A HEURISTIC ALGORITHM FOR AN INTEGRATED ROUTING AND SCHEDULING PROBLEM WITH STOPS EN-ROUTE

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Emre Uzun

May, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Osman Alp (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Oya Karaşan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Haldun Süral

Approved for the Institute of Engineering and Science:

Prof. Mehmet B. Baray
Director of the Institute

ABSTRACT

A HEURISTIC ALGORITHM FOR AN INTEGRATED ROUTING AND SCHEDULING PROBLEM WITH STOPS EN-ROUTE

Emre Uzun
M.S. in Industrial Engineering
Supervisor: Asst. Prof. Osman Alp
May, 2009

In this study, we examine an integrated routing and scheduling problem that arises in the context of transportation of hazardous materials. The purpose of the problem is to find a minimum risk route between an origin and a destination point on a given network and to build a schedule on this route that determines where and how long to stop for a truck carrying hazardous materials. The objective is to minimize the risk imposed to the society while completing the path within a given time limit. The risk is defined as the expected population exposure in the presence of an accident which varies different times in a day. There are exact algorithms available in the literature that solve the problem. However, these algorithms are not capable of solving large sized networks due to memory constraints. Our aim is to develop a heuristic procedure that can handle larger networks. We separate the problem into two independent components, routing and scheduling, and propose solution algorithms which would communicate each other when running the algorithm. For the routing component we define a neighborhood structure that can be used to generate several paths around a given path on a network. The search procedure takes an initial path and improves it by generating different paths in the defined neighborhood. For the scheduling component, we discuss mixed integer programming, dynamic programming and heuristic approaches. We run the proposed heuristic algorithm on several test networks and compare its performance with the optimal solutions. We also present the application of the heuristic procedure on a large sized Turkey Road Network.

Keywords: Integrated Routing and Scheduling Problem, Heuristic Procedures, Dynamic Programming.

ÖZET

YOL ÜZERİNDE DURMAYI DİKKATE ALAN BÜTÜNLEŞİK ROTALAMA VE ÇİZELGELEME PROBLEMLERİ İÇİN SEZGİSEL BİR YÖNTEM

Emre Uzun
Endüstri Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Yrd. Doç. Dr. Osman Alp
Mayıs, 2009

Bu çalışmada karayollarında tehlikeli madde taşıyan araçlarda sıkça karşılaşılan bütünleşik rotalama ve çizelgeleme problemi incelenmiştir. Problemin amacı, bir ağ üzerindeki iki nokta arasında en düşük riski veren bir rota ve bu rota üzerinde seyreden tehlikeli madde taşıyan bir aracın nerede ve ne kadar durması gerektiğini belirten bir çizelge bulmaktır. Problemin hedefi, rota üzerinde herhangi bir kaza durumunda etkilenecek ve gün içerisinde değişen ortalama insan sayısını en aza indirgeyecek, önceden belirlenmiş bir süre içerisinde kat edilmesi gereken rotayı ve çizelgeyi belirlemektir. Literatürde bu problemi optimal olarak çözen algoritmalar mevcuttur. Ancak bu algoritmalar büyük boyutlu ağlarda yetersiz kalmaktadır. Bu tezdeki amaç, büyük boyutlu ağlarda çalışabilecek bir sezgisel yöntem geliştirmektir. Problemin rotalama ve çizelgeleme süreçleri, birbirleri ile iletişim halinde olan iki ayri süreç olarak belirlenip, bunlar için farklı çözüm yöntemleri geliştirilmiştir. Rotalama için her iterasyonda bir önceki iterasyonun en iyi rotasını kullanarak yeni rotalar üreten bir sezgisel yöntem üzerinde durulmuştur. Çizelgeleme için ise karışık tamsayılı programlama, dinamik programlama ve sezgisel yöntemler olmak üzere üç ayrı yaklaşım tartışılmıştır. Geliştirilen algoritma çeşitli test ağlarında uygulanmış ve performansı optimal sonuçlar ile karşılaştırılmıştır. Ayrıca, Türkiye Karayolları Ağı kullanılarak algoritmanın büyük ağlardaki performansı test edilmiştir.

Anahtar sözcükler: Entegre Rotalama ve Çizelgeleme Problemi, Sezgisel Yöntemler, Dinamik Programlama.

Acknowledgement

First and foremost, I would like to express my deepest and most sincere gratitude to my supervisor Asst. Prof. Osman Alp for his invaluable guidance, encouragement and motivation during my graduate study. He has been always ready to provide help, support and trust with everlasting patience and interest. I have learned a lot of things from him, not only in academic but also in personal and intellectual matters.

I would like to thank to Assoc. Prof. Oya Karaşan and Assoc. Prof. Haldun Süral for accepting to read and review this thesis and their substantial comments and suggestions.

I would like to thank my officemates Nurdan Ahat, Yahya Saleh, Tuğçe Akbaş, Esra Koca, Ece Z. Demirci, Işık Öztürkeri and Ersin Körpeoğlu for their invaluable patience and kindness during my graduate study since it should have been very hard for them to cope with me at certain times. It is a pleasure for me to express my deepest gratitude to my friend Sıtkı Gülten for being so patient, helpful and considerate during my graduate study. I also wish to thank Hatice Çalık for being ready to listen to me carefully and sharing her valuable thoughts with me all the time. I learned a lot from you.

I am indebted to Can Öz, Efe Burak Bozkaya and Onur Özkök for their valuable support during my graduate study.

I want to thank my classmates Safa Onur Bingöl, Merve Çelen, A. Burak Paç, Könül Bayramoğlu, Yüce Çınar, G. Didem Batur, Burak Ayar, Serdar Yıldız, Zeynep Aydın and Muzaffer Mısırcı for being so considerate and gracious.

I am grateful to Füsun Şahin, Gökçe Akın, Ahmet Korhan Aras, İhsan Yanıkoğlu, Barış Cem Şal, Gülşah Hançerlioğulları, Yiğit Saç, Karca Duru Aral, Pelin Damcı, Utku Guruşçu and all other colleagues for providing me such a friendly environment to work and also for their entertaining chats and lunch breaks when I want to abstain from doing work.

I also wish to thank “Kaytarıkçılar” Esra Aybar, M. Fazıl Paç, Gülay Samatlı, İpek Keleş, F. Safa Erenay, Nasuh Çağdaş Büyükkaramıklı, Mehmet Mustafa Tanrıkulu and Erdinç Mert for their comradeship in the very early days of my graduate study. I would like to thank Ahmet Camcı, Sibel Alumur, Önder Bulut, Ayşegül Altın, Utku Koç and all of the friends that I failed to mention here for their friendship and support during my graduate study.

I indebted to my dear friends Muratcan Alkan, Fırat Karataş, Mustafa Ersoy, Alper Kargı for their morale support and keen friendship.

Also, I would like to express my gratitude to TÜBİTAK for its financial support throughout my Master’s study.

Finally, I would like to express my deepest gratitude to my family for their everlasting love and support.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Transportation of Hazardous Materials (hazmats) is an important issue that receives public attention due to high risk threats on the society, since the consequences of an accident involving a hazmat truck may be catastrophic due to explosions or poisonous spills. Some of the example hazmats are flammable liquids, radioactive materials and poisonous gases. Various different modes of transportation such as air, water, highway, rail and pipeline are used to transport these materials. Highways, or more specifically, truck usage is significantly higher than the others in hazmat transportation. According to the statistics of United States Department of Transportation (DOT) [27], 52.9% of hazmat transportation in weight is handled by trucks which is followed by pipelines with 20%, ships with 10%, trains with 5% and other 12.1%. There are certain advantages of using trucks. They are more flexible to reach many locations than the other modes and the costs associated with using trucks can be lower when small amounts of materials are transported. However, there is no stringent central control over traffic, route selection, behavior, etc. in highway transportation (except some regulations that will be explained later) as the drivers and the carrier companies are free to set their transportation patterns whereas, vehicles like airplanes, ships are centrally controlled by computerized tools that regulate the traffic to increase safety measures. This situation takes considerable attention of governments, insurance companies, social organizations and public because trucks are involved in 85%

of all hazmat related accidents. Table 1.1 shows the hazmat accident statistics obtained from DOT [27] between years 1998 and 2007.

Table 1.1: The Incidents and the Consequences of Hazmat Vehicles Between 1998 and 2007 in USA

Mode	**Incidents**	**Fatalities**	**Injuries**	**Damage ($)**
Air	35	0	0	152,984
Highway	3127	96	194	339,062,628
Rail	509	14	747	157,260,628
Water	0	0	0	0
Total	3671	110	941	496,476,240

Although, the percentage of hazmat truck accidents constitutes a very small percentage of all of the traffic accidents, the consequences associated are significantly higher than those of an ordinary accident. The release of the materials that a hazmat truck carry can seriously harm people in other vehicles around, as well as people living nearby and the environment. Depending on the released material, there could be a persistent damage that may remain for many years, especially in radioactive cases.

The risk of an accident cannot be neglected in the presence of other vehicles and people around. Developed countries like USA enforce strict regulations and restrictions to decrease this risk to "one in one million". In developing countries the conditions are inferior. In Turkey, more than 44 people died and 236 injured in hazmat related traffic accidents between 2005 and 2008 [1]. Regarding these consequences, the hazmat transportation planning should carefully be done.

There are certain components of a typical hazmat transportation planning, namely, routing and scheduling. First of all, a route from an origin to the destination should be determined for the vehicle. Erkut and Verter [9] discuss different approaches proposed in literature to find routes for hazmat vehicles. Use of single objective models that select minimum risk path is the most common method

[1] Statistics gathered from the hazmat related traffic accident news published between years 2005 and 2008. Obtained from the Hürriyet Newspaper Archives (http://hurarsiv.hurriyet.com.tr/arsiv/).

implemented. However, the problem is also modeled as a multi-criteria optimization problem. While some authors consider two criteria in which, distance and population exposure objectives were used, some defined three criteria where population exposure, accident likelihood and operating costs are minimized and some defined special population groups (for evacuation reasons) as the fourth criteria in addition to the previous three. Erkut and Verter [9] mention about the development in information technology that leads to the usage of detailed databases about the road conditions and population counts and the geographical information systems in route selection process. This development provides consideration of many additional criteria while selecting routes in addition to the previously stated.

Time-dependency is another important issue to be considered in routing. Certain criteria such as accident risks or population exposures can be time-dependent. For instance, the cyclic population movements affect these measures since the risk of passing through a city in the rush hours is significantly higher. Stopping enroute is another factor to be considered. If stopping is allowed, a vehicle arriving at a city during the rush hour can stop and wait till the end of the rush hour in order to decrease the risk. The governmental policies sometimes restrict trucks carrying hazardous materials to pass through some cities, bridges or tunnels in certain times of the day. This restriction which causes the truck to have an undesired delay at certain points results in an increase in the operating costs.

Schweitzer [19] mentions about the significance of human factors in hazmat accidents. This highlights the importance of driving conditions of the truck drivers. Most countries have specific driving time restrictions that apply for all commercial vehicle drivers. For instance in USA, effective from October 2005, Department of Transportation defines the following regulation:

- Drivers cannot continuously drive more than 11 hours in an 14 hour of on-duty time.
- Drivers cannot be on duty more than 14 hours including the short breaks taken.

- Drivers should take a break no less than 10 hours after 14 hours of on duty period.

European Union [24] also imposes certain regulations:

- Drivers must take a break no less than 45 minutes after driving 4.5 hours. This break may be separated into breaks of size 15 and 30 minutes and distributed among the 4.5 hours driving period.
- Drivers are not allowed to drive a total of 9 hours within a 24 hour period.
- Drivers should take an uninterrupted break no less than 11 hours in every working day or this break can be divided into an uninterrupted break of 9 and 3 hours.

The scheduling component of hazmat transportation deals with finding a driving schedule on a given route. This schedule provides driving and stoppage durations en-route. All the factors mentioned above should be considered when building realistic driving schedules.

The problem that we consider in this thesis is an integrated routing and scheduling problem. In particular, the aim is to find a route between an origin and a destination on a given network; and a driving schedule on this route providing when and how long to drive on the route and when and how long to stop en-route, so that the selected risk and/or cost measures are optimized. We consider time dependent risk measures. The vehicles are allowed to stop at any node on the path. An upper bound on the total trip time is defined so that the objective of the problem is to find the path that yields the minimum risk within the total trip time limit.

The definition of the risk measure is a core component of the problem, since routing and scheduling decisions depend on how the risk measure is defined. Although different points of view have been developed to calculate the risk of an accident in the literature, there is no consensus on how the risk should be defined.

According to the surveys by List et. al. [14] and Erkut and Verter [9], some authors define risk as a composition of three stages: probability associated with an undesirable event (a hazmat accident with a release), determination of the level of potential population exposure and property exposure and magnitude of consequences given the level of exposure. On the other hand, approaches to deal with the time of the day (day vs. night) and the road (dry vs. wet) conditions or the expected consequence given the occurrence of the first accident (conditional risk) and duration of exposure (perceived risk) have also been considered in previous research. For other possible measures see List et. al. [14] and Erkut and Verter [9].

In this study, the risk measure is defined as the expected number of people exposed to risk that is caused by the movement of hazmat trucks along a path. The expected number of people exposed to risk along an arc can be calculated by multiplying the probability of a hazmat accident by the number of people exposed to the damage that will be caused by the release of the hazmat, an explosion, and the like when the accident occurs. Erkut and Verter [10] mention that the population to be exposed lies within a danger circle with radius r, whose origin is a point on the road segment as illustrated in Figure 1.1 and r is a given parameter depending on the nature of hazmat (gas, liquid, explosive, etc.). The risk of a path is simply calculated by the summation of the risk measures associated with each arc on this path. This risk measure is commonly used in related literature. Moreover, these risk measures can be time dependent since the probability of an accident and the amount of people to be exposed may change during the day.

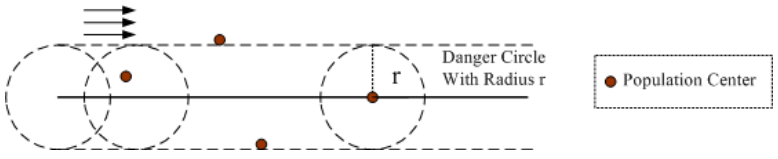


Figure 1.1: Illustration of Impact Radius

The problem that we are working on is closely related to the constrained shortest path problem with stops en-route. The objective of the problem is to minimize the risk of the expected exposure in case of an accident, which is a time varying parameter. There have been many different approaches developed in the

literature for the constrained shortest path problem. Joksch [13] proposes a linear programming and a dynamic programming formulation to the problem and compares the difficulties of two methods. Aneja and Nair [3] treat the problem as a bi-criteria optimization problem and propose a linear programming model along with a labeling algorithm that finds extreme non-dominated solutions to the LP formulation. Handler and Zang [12] develop an algorithm to close the gap between the Lagrangian dual of the LP formulation of constrained shortest path problem and the solutions found by using the k^{th} shortest path problem. Riberio and Minoux [17] propose a pseudo-polynomial algorithm based on Lagrangian dual that generates solutions for the constrained shortest path problems with double sided inequality constraints. Dumitrescu and Boland [7] provide an improvement technique to label setting method by the bounds obtained from Lagrangian relaxation in the preprocessing stage. Santos et. al. [18] develop a new solution strategy for the constrained shortest path problem that utilizes the k^{th} shortest path problem by defining an efficient search direction. Zhu et. al. [20] develop a pseudo-polynomial three stage algorithm that is composed of a preprocessing stage, an expansion stage and an optimization stage that mainly utilizes column generation to solve each sub-problem instance. All of the previous studies about constrained shortest path problem mentioned above does not consider time-dependent arc attributes. However, our problem definition requires time dependent risk attributes.

The concept of time-dependent arc attributes is first introduced by Cooke and Halsey [5]. They propose a procedure based on Bellman's algorithm to find the shortest path on a given network whose arc attributes are a function of the starting time to traverse an arc. Halpern [11], develops an extension to the Dijkstra's algorithm that is capable of solving the same problem. This approach also works with cases where waiting at certain nodes are limited or prohibited. Orda and Rom [16] generalize the work done by Cooke and Halsey [5] and Halpern [11] and develop an algorithm that is capable of solving different cases of the time-dependent shortest path problem: (1) unlimited waiting at each node, (2) no waiting is allowed, and (3) waiting is only allowed at the origin. Cai et. al. [4] develop an algorithm to solve the time-dependent shortest path problem with an

upper bound on the total trip time. Three different approaches to the problem are presented: (1) arbitrary waiting times at nodes, (2) zero waiting times, and (3) bounded waiting times. The authors develop a pseudo-polynomial algorithm based on dynamic programming that solves each case optimally.

An integrated routing and scheduling approach is described by Nozick et. al. [15] who propose a time dependent extension to the labeling algorithm developed by Cox [6] that is capable of solving multi-objective routing problems. Given a starting time of the trip, the admissible nodes are labeled with the performance measure of the non-dominated routes at each iteration and the algorithm terminates with the non-dominated routes from origin and destination. The objectives of the model is to minimize the route length, total population exposure and the total accident probability. The authors also present a case study that covers a portion of a state in USA.

Later, Erkut and Alp [8] extended this work by allowing vehicle to stop at any node during the trip. The objective of the problem however, is defined as to minimize one arc attribute (exposure or accident probability or risk) rather than a multi-objective minimization. They handle the problem in four different cases each time restricting one aspect of the model: unrestricted waiting and driving times, restricted waiting and unrestricted driving times, restricted driving and waiting times and complex restrictions on waiting and driving times in which the United States Department of Transportation regulations effective in the year 2003 are implemented. They propose a mixed integer programming and a dynamic programming formulation to each case and they perform computational tests of each case on the network used by Nozick et. al. [15] and compare the results.

The fourth dynamic programming model proposed by Erkut and Alp [8] solves our problem optimally, but it fails to solve larger sized networks because of the huge memory requirements. For instance, around 1 GB of memory is required to handle a portion of a state in USA which has 138 nodes and 368 arcs. Our primary purpose is to find solutions for larger sized networks. Therefore we develop a heuristic procedure to achieve this objective. We use the same parameter and policy settings used by Erkut and Alp [8] in order to have a basis for comparison.

The settings that we use in our problem are described in the following paragraphs:

The driving and on duty time restrictions that Erkut and Alp [8] used in their paper are as follows: Two types of break are defined. (1) Short break is the break that can be at most eight hours. Examples to a short break can be a lunch break or a fueling break. Moreover, short breaks can also be used when the vehicle stops before a city and waits for the end of the rush hour period in order to decrease the risk of an accident. (2) Long break is the break that is at least eight hours and they are intended for resting. Two types of working period are defined. (1) Uninterrupted Driving Time is the period in which the driver continuously drive the truck without any breaks. (2) On-Duty Time is the period which contains driving period as well as short breaks but not the long breaks that are basically used for resting. Then, under these definitions the following regulations are applied to the truck drivers:

- Uninterrupted Driving Time cannot exceed 10 hours, whenever it reaches 10 hours a long break must be given.
- Drivers cannot be On-Duty more than 15 hours, a long break must be given.
- Drivers must give a break no less than 8 hours after a maximum of 15 hours of on duty period.

There is an upper bound on the total trip time since the lack of this restriction can result a situation that the vehicle stops at each node and waits for a sufficient amount of time so that the next arc is traversed with minimum risk.

The rest of the thesis is organized as follows. In the next chapter the detailed description of the heuristic is given. The method to build routes and generate neighborhoods of the heuristic algorithm is given. Different approaches to build a feasible schedule are proposed and compared. In the third chapter, the proposed heuristic is implemented on various different networks and the results are compared with the ones that are obtained from the dynamic programming formulation of [8]. In the fourth chapter, we present the performance of our heuristic procedure on a large sized network.

Chapter 2

Methodology

In this chapter, we present our methodology to solve the problem. This is a heuristic approach where we decompose the problem into two components: a routing component and a scheduling / risk estimation component. We generate several routes connecting the source and destination nodes in an iterative fashion and solve a subproblem for each route which estimates the risk by finding a driving schedule on that route. The driving schedule indicates the time periods in which the vehicle stops at a particular node. We give our notation, parameters and assumptions in the next section and then we define our main algorithm. Afterwards, we present our approaches to handle the basic steps of the algorithm such as generating routes, neighborhoods, and performing the scheduling operation.

2.1 Notation, Parameters and Assumptions

We have a directed graph $G = (N, A)$ where N is the set of nodes which represent population centers or highway intersection points and A is the set of arcs which represent highway segments. The arcs have time dependent risk and time independent traversing time attributes. We define a directed path $P = (N^P, A^P)$, where $P \subseteq G$ is a path on graph G that initiates from node $s \in N^P$ and terminates at node $d \in N^P$. The sets $N^P \subseteq N$ and $A^P \subseteq A$ are defined as the node

Table 2.1: Time Related Problem Parameters

Parameter	**Explanation**
W_{MAX}	Maximum amount of on-duty time periods
D_{MAX}	Maximum amount of driving time periods
L_{MIN}	Minimum amount of time required for a long break
S_{MAX}	Maximum amount of time required for a short break
T_{MAX}	Maximum amount of time required for the trip

and arc sets of the path P, respectively. We define a function $R(P)$ which returns the risk associated with path P. We also define R_{tij} as the risk of entering arc $(i,j) \in A$ at time t and d_{ij} as the distance in terms of traversing time of arc $(i,j) \in A$. Our problem has five time related parameters which are given on Table 2.1. Our aim is to find a path P from a source node s to a destination node d such that $R(P) = min\{R(\overline{P}) : \overline{P}$ is a path from s to d in $G\}$ on which the total duration of the trip, the total driving time, and the total on-duty time is no more than T_{MAX}, D_{MAX}, and W_{MAX} respectively with each short break being no more than S_{MAX} and each long break being no less than L_{MIN}.

We make the following assumptions in our problem:

- The risk of traversing an arc is determined by the risk at the time the vehicle starts traversing it.
- The speed of the vehicle and the duration of traversing an arc is constant and independent of the time of the day.
- The vehicle is not allowed to stop on an arc.
- The vehicle is safe to stop at a node with no risk.
- The time is discretized into small units.

2.2 Main Heuristic Procedure

The algorithm that we propose is basically an improvement heuristic procedure that takes an initial path as an input and outputs the best path with its driving schedule. At each iteration, paths in the "neighborhood" of a path P, which is the "best" path of the previous iteration, is generated and the risks associated with these new paths are calculated. The path P' that yields the best risk among the paths in the neighborhood is recorded and the neighborhood of P' is generated in the next iteration. The algorithm works in this manner until no improving solutions exist in the neighborhood. The outline of the procedure is given in Algorithm 1.

Algorithm 1 Outline of the Heuristic Procedure

```
Initialize
while There exists an improving solution in the neighborhood of current path
do
    Generate new paths from the neighborhood of the current path
    Calculate the risks associated with each generated path in the neighborhood
    Record the path with the best risk
end while
Output the best path obtained
```

In our original problem, scheduling is integrated with routing, and routing is dynamically affected by scheduling. As can be inferred from Algorithm 1, we separate the routing and scheduling operations, since the time dependent arc attributes complicates the route selection procedure. Obviously, building a schedule for a given path is easier than an integrated modeling. The routing module involves the main heuristic steps where neighborhood generation and route selection processes take place. On the other hand, the scheduling module calculates the risk associated to a path, so it can be viewed as a performance evaluator module. The key point of this approach is the modular structure. Whenever a path is generated, calculating the risk of this path is totally independent of how it is generated. Likewise, risk calculation does not affect the process of generating paths. Thus, the problem can be viewed as a composition of two distinct and separable modules that communicate each other as described in Figure 2.1.

This unique property of the problem provides flexibility to substitute a module without changing the whole process.

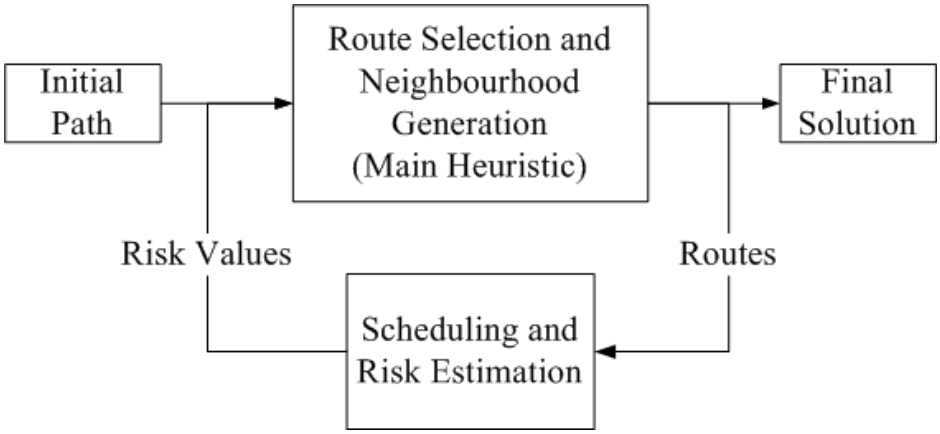


Figure 2.1: The General Structure of the Algorithm

The next sections of this chapter are devoted to the detailed explanations of the components described in Figure 2.1.

2.3 Initialization

In the initialization part of the of the algorithm, the initial path of the heuristic is constructed. Given source and destination nodes $s, t \in N$, an initial path can be found by tracing the Breadth-First Search tree rooted at node s. See Appendix A for the pseudocode of the Breadth-First Search algorithm.

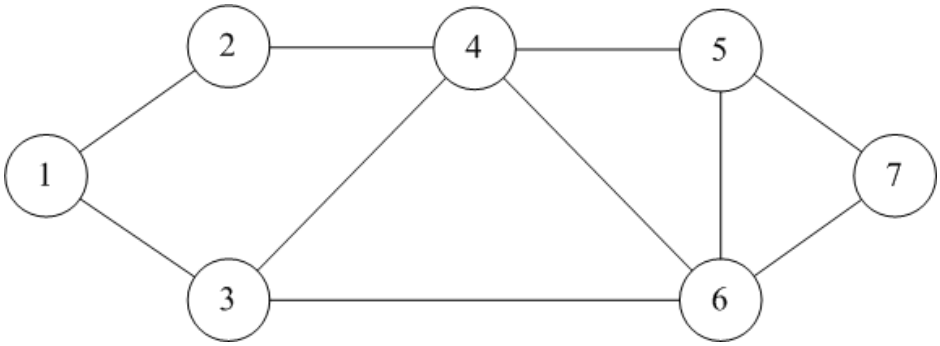


Figure 2.2: A Sample Network

Suppose that we have a network given in Figure 2.2 where, nodes 1 and 7 are the origin and destination nodes, respectively. The Breadth-First Search Algorithm rooted at node 1 terminates with a tree shown in Figure 2.3. Then, the path from nodes 1 and 7 is 1 - 3 - 6 - 7.

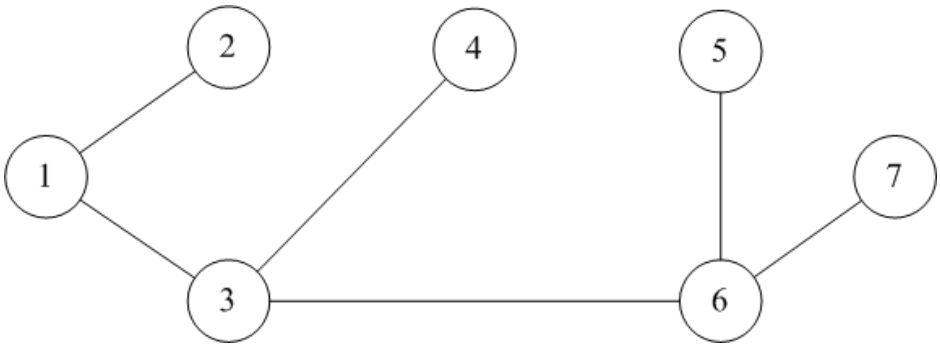


Figure 2.3: The Breadth-First Search Tree Initiated from node 1

2.4 Route Selection and Neighborhood Generation

Given a graph $G = (N, A)$, the primary purpose of this module is to generate different paths for the Scheduling and Risk Estimation module using the prior information obtained from the same module.

2.4.1 Neighborhood

Since our algorithm is an improvement heuristic, a neighborhood of a path P should be generated at each iteration in order to move to a new path P'. We need to consider many different aspects while generating a neighborhood. The problem parameters such as time dependent risk attributes or the time restrictions have significant effects on the optimal path. For instance when T_{MAX} is small, the optimal path may be one of the paths that are close to the shortest path in terms of traversing time. On the other hand, for larger values of T_{MAX}, the optimal path may move away from the shortest path. Thus, the neighborhood

should include different paths that span a wide range of the network. Before defining our neighborhood structure, we make the following definition:

Definition 1 *Let $P = (N^P, A^P)$ be a given path. A Partial Path $PP = (N^{PP}, A^{PP})$, $PP \subseteq G$, $N^{PP} \subseteq N$, $A^{PP} \subseteq A$ is defined as any path initiated from node $n \in N \backslash N^P$ and terminated at node $b \in N^P$ and $N^{PP} \cap N^P = \{b\}$. The set of partial paths of node n for a given path, SPP_n, is defined to contain all of the partial paths that are found by using Breadth-First Search tree initiated from node n.*

The usage of Breadth-First SFearch tree implies that no two distinct partial paths from node n terminate at the same node on the path P, since once a node is reached by an arc and marked at any iteration of the Breadth-First Search, it can no longer be reached by another arc. Moreover, the partial paths obtained are the shortest paths in terms of arcs.

For instance, in the network shown in Figure 2.4, where the path P is represented as s = 1 - 3 - 6 - 7 = d. The partial paths $PP \in SPP_4$ of node 4 are: 4 - 2 - 1, 4 - 3, 4 - 6 and 4 - 5 - 7. The path 4 - 5 - 6 is not a partial path of node 4, since node 6 belongs to partial path 4 - 6.

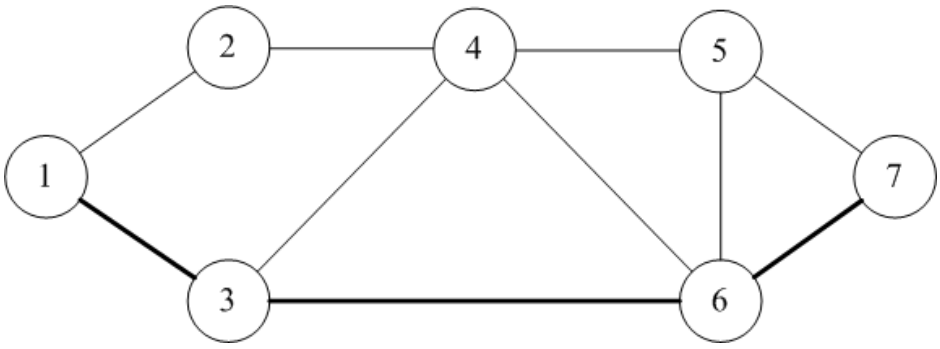


Figure 2.4: Partial Paths Initiated From Node 4

Definition 2 *Let $P \subseteq G$ be a path started from node $s \in N^P$ and ended at $t \in N^P$; $n \in N$ be a node such that $n \notin N^P$ and PP_1 and PP_2 be two distinct partial paths initiated from node n and terminated at nodes $b_1 \in N^P$ and $b_2 \in N^P$,*

respectively, such that b_1 comes before b_2 on P. A path $P' \subseteq G$, $P' \neq P$ that is constructed by linking path segments (s, b_1), (b_1, n), (n, b_2) and (b_2, t) is called a neighboring path.

The neighboring paths of node n is all of the possible combinations of all partial paths PP such that $PP \in SPP_n$. In the example given in Figure 2.4, path 1 - 2 - 4 - 6 - 7 is a neighboring path generated from the partial paths 1 - 2 - 4 and 4 - 6 of node 4.

Definition 3 *The neighborhood of a given path $P = (N^P, A^P)$ is defined as the collection of the neighboring paths obtained from all nodes n such that $n \in N \backslash N^P$.*

The neighborhood of path P given at Figure 2.4 contains the following paths:
1 - 2 - 4 - 3 - 6 - 7
1 - 2 - 4 - 6 - 7
1 - 2 - 4 - 5 - 7
1 - 3 - 4 - 6 - 7
1 - 3 - 4 - 5 - 7
1 - 3 - 6 - 4 - 5 - 7
1 - 2 - 4 - 5 - 6 - 7
1 - 3 - 4 - 5 - 6 - 7
1 - 3 - 6 - 5 - 7

Since the number of Partial Paths PP obtained from node n can be at most $\begin{pmatrix} |N^P| \\ 2 \end{pmatrix}$, total number of paths that can be generated at any iteration of the algorithm can be theoretically at most $|N \backslash N^P| \begin{pmatrix} |N^P| \\ 2 \end{pmatrix}$, which is $O(|N|^3)$. However, with the Neighborhood Contraction methods which are introduced in the next section; the actual realization of the number of paths generated is much less than this number in our proposed algorithm.

In Chapter 4, we present the application of our heuristic procedure on Turkey Road Network. In Figure 2.5, we visualize three different neighboring paths for a node $n \notin N^P$ that are constructed in one iteration of our procedure.

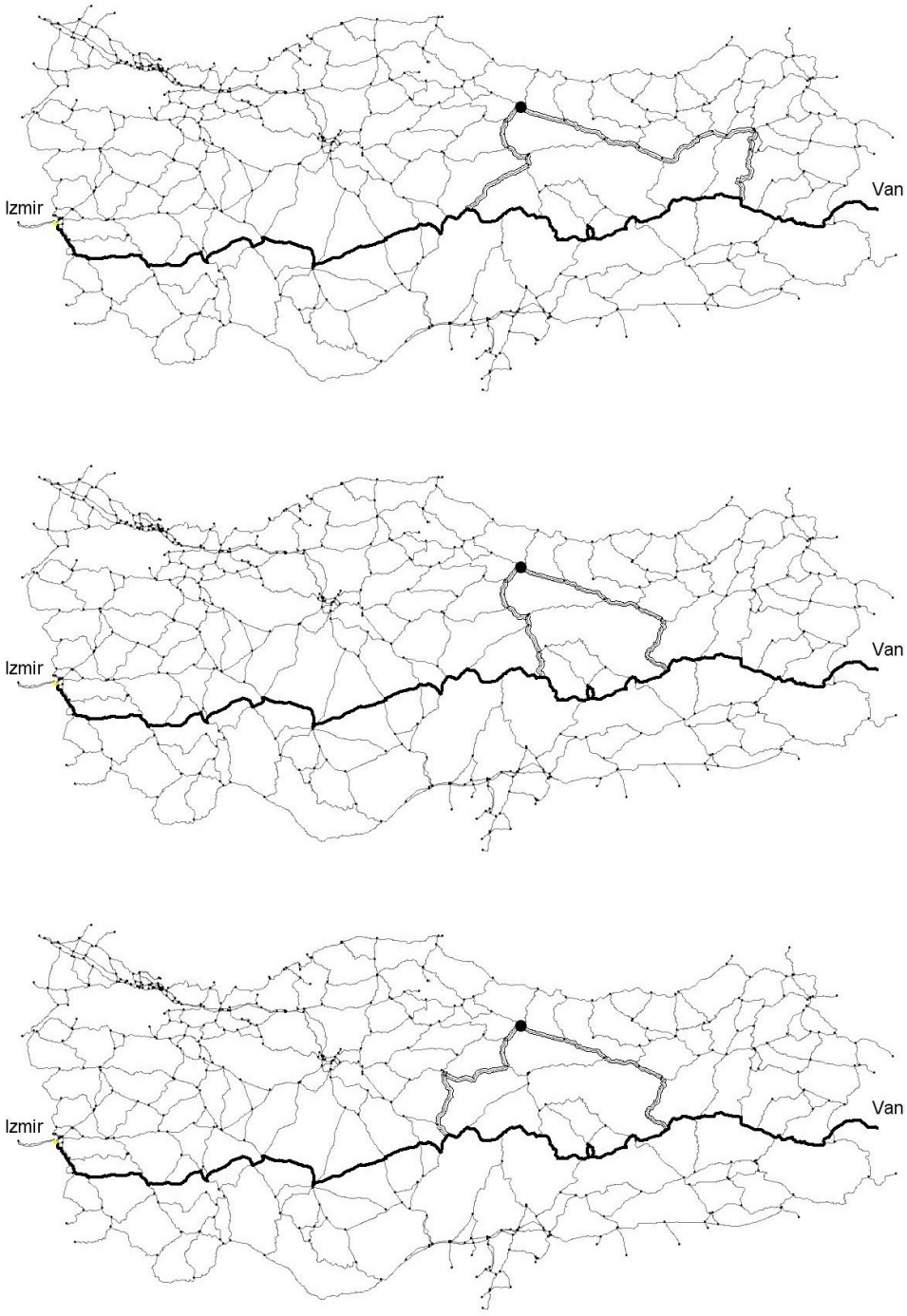


Figure 2.5: An Example to the Neighboring Paths Constructed in an Iteration of the Algorithm

2.4.2 Neighborhood Contraction

The neighborhood structure defined at Section 2.4.1 generates a large number of paths to span the network. However, some of these paths, especially the ones with high risk values, may not likely to be the optimal one. The elimination of these paths will shrink the size of the neighborhood, hence speeds up the algorithm. We develop two neighborhood contraction methods that depend on the statistical occurrence of the nodes in each iteration. Before defining these methods, we introduce the following sets:

Recall that a neighborhood of a path P is the collection of all neighboring paths of all nodes $n \in N \backslash N^P$. We define the Set of Best Paths of Neighborhood (SBP_N) as the collection of best neighboring paths of each node $n \in N \backslash N^P$. We call the path with the minimum risk in the set SBP_N as the Best Path of Iteration (BP_I). If a path P' is a BP_I then it is also in the Set of Best Paths of Iterations (SBP_I). We also keep a Set of Best Paths of Algorithm (SBP_A) which contains at most k paths, which has the first k smallest risk values among all of the paths calculated throughout the algorithm, where k is a given algorithm parameter.

The sets SBP_N, SBP_I and SBP_A may have a predefined and fixed size defined as S_{SBP_N}, S_{SBP_I}, S_{SBP_A} respectively, so that a new path P can only enter if
$R(P) < max\{R(\overline{P}) : \overline{P} \in SBP_N\}$ or $|SBP_N| < S_{SBP_N}$ for SBP_N
$R(P) < max\{R(\overline{P}) : \overline{P} \in SBP_I\}$ or $|SBP_I| < S_{SBP_I}$ for SBP_I
$R(P) < max\{R(\overline{P}) : \overline{P} \in SBP_A\}$ or $|SBP_A| < S_{SBP_A}$ for SBP_A

Definition 4 *The appearance $A(n)$ of node n in a set of paths S is defined as the number of paths $P \in S$ such that $n \in N^P$. The node n is said to be a "significantly appearing node" if $A(n) \geq \lambda max\{A(m) : m \in N^P \text{ and } P \in S\}$ where $0 \leq \lambda \leq 1$ is a given algorithm parameter.*

Definition 5 *Let $P = (N^P, A^P)$ a given path. Node $n \in N \backslash N^P$ is called a Promising Node (PN), if node n has a significant appearance in SBP_N when*

running the algorithm. Node $n \in N \backslash N^P$ is called a Non-Promising Node (NPN), if it does not have significant appearance.

The Non-Promising Nodes are nodes whose neighboring paths are all infeasible or have relatively higher risk values.

Neighborhood Contraction Methods are developed to eliminate non-promising nodes gradually throughout the algorithm to shrink the size of the neighborhood and to direct the search procedure generate new paths around promising nodes. Window Shifting and Cumulative Occurrence methods are developed to achieve this objective. Either one of these methods could be implemented in the algorithm.

2.4.2.1 Window Shifting (WS)

The neighboring paths of node $n \in N \backslash N^P$ at iteration i will be constructed if, node n occurs in one of the best path of iterations in the previous w iterations where w is a given algorithm parameter. The method does not restrict any node in the first w iterations.

2.4.2.2 Cumulative Occurrence (CO)

The neighboring paths of node $n \in N \backslash N^P$ at iteration i will be constructed if, node n is a promising node. In order to collect statistical data, the method does not restrict any node for a predefined warm up period ϵ in the beginning.

2.4.2.3 Rescheduling Avoidance

A path P' is discarded, if it has already been generated in the same neighborhood or in the previous neighborhoods.

Algorithm 2 Route Selection and Neighborhood Generation Heuristic

$SBP_I := \emptyset$ and $SBP_A := \emptyset$
Take initial path P, Set $SPB_I = \{P\}$ and $BP_I = P$
while There is an untraced path $P \in SPB_I$ **do**
  $SBP_N := \emptyset$
  for all $a \in N$ such that $a \notin N^P$ **do**
    if (a is a PN and CO is used) OR (a occurs in one of the BP_I in the previous w iterations and WS is used) **then**
      Generate the set SPP_a
      for all possible paths P' obtained from combinations of $PP \in SPP_a$ **do**
        if P' has never been scheduled **then**
          Build Schedule and Calculate Risk
          if P' can enter SBP_N **then**
            Perform necessary operations and set $P' \in SBP_N$
          end if
          if P' can enter SBP_I **then**
            Perform necessary operations and set $P' \in SBP_I$
          end if
          if P' can enter SBP_A **then**
            Perform necessary operations and set $P' \in SBP_A$
          end if
        end if
      end for
    end if
  end for
end while

2.4.3 Procedure

The algorithm for Route Selection and Neighborhood Generation which is the detailed version of Algorithm 1, starts with an initial path P obtained from the initialization part of the algorithm. The algorithm then generates the neighborhood of the path P by using the partial paths obtained from nodes that are not on the path regarding the neighborhood contraction methods. Afterwards, for each path P' in the neighborhood, the associated risk $R(P')$ is calculated. If the path P' can enter the sets SBP_N, SBP_I, SBP_A, the elements of these sets are updated. When all of the neighborhood of path P is traced (this is the time one iteration finishes), the algorithm picks the path P^* that yields the minimum risk

in the SBP_N and sets it as the current path and the procedure continues with the neighborhood of path P^*. The procedure terminates when no new paths remain in the set SBP_I. The path that yields the minimum risk in the set SBP_I is the result of the procedure. The pseudo-code of this procedure is given as Algorithm 2.

2.5 Scheduling and Risk Estimation

In this module, we solve the scheduling / risk estimation subproblem for each route generated by the Route Selection and Neighborhood Generation module. This subproblem seeks to find the best driving schedule that yields the minimum risk on a single path. As discussed before, the schedule identifies the amount of time the vehicle stops at each node. Given an upper bound T_{MAX} on the total trip time and T_P which is the total traversing time without any breaks, building a schedule can be defined as the allocation of a maximum of $T_{MAX} - T_P$ amount time to the nodes.

This can be further explained by a simple example. In Figure 2.6, let 1 - 3 - 6 - 7 be the path to be scheduled with traversing times three, five and eight minutes for arcs (1,3), (3,6) and (6,7), respectively and $T_{MAX} = 20$ minutes are available for the whole trip. Then, the total traversing (driving) time without any breaks is, $T_P = 3 + 5 + 8 = 16$ minutes and $20 - 16 = 4$ minutes remain to be distributed as breaks. Then, in order to find the driving schedule that yields the minimum risk, up to four minutes of break can be distributed to nodes 1, 3 and 6. Notice that if the schedule without any breaks is feasible and gives the minimum risk, no breaks may be distributed at all. If we do not allocate any breaks to any of the nodes, then the total risk of this path will be the summation of the risks of traversing arc (1,3) between times 0 and 3, traversing the arc (3,5) between times 3 and 8, and traversing the arc (6,7) between times 8 and 16. In such a case, the total trip duration (and the total driving time) will be simply 16. However, if we insert 2-minute breaks at nodes 3 and 6 for example, the total risk of this path will be the summation of the risks of traversing arc (1,3) between times 0 and

3, traversing the arc (3,5) between times 5 and 10, and traversing the arc (6,7) between times 12 and 20. In such a case, the total driving time will still be 16 but the total trip duration will be 20. Due to the time dependent risk measures, the total risk values of the above examples could be different.

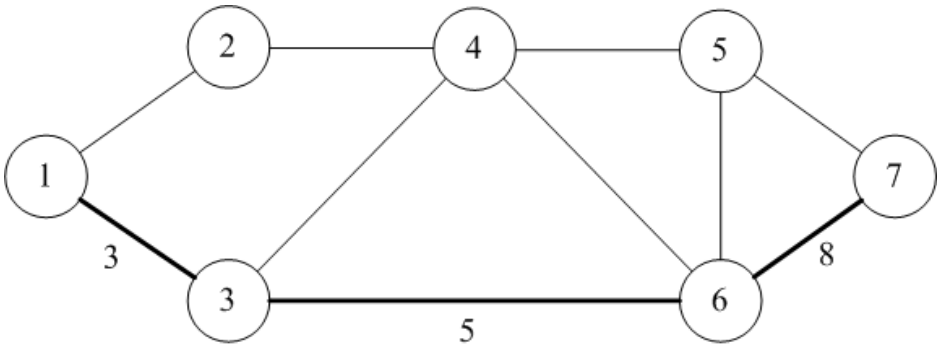


Figure 2.6: A Sample Path on a Network

This subproblem can be solved by numerous methods. We discuss two exact formulations (MIP and DP) and a heuristic algorithm to solve this subproblem.

2.5.1 Integer Programming Approach

The problem of constructing a schedule to a path can be viewed as a Resource Allocation problem, since the breaks are distributed among the nodes. However, due to the time dependent nature of the problem, we cannot model it as a typical Resource Allocation Problem. We provide a Mixed Integer Programming formulation (MIP-1).

2.5.1.1 Decision Variables

We need to define the following decision variables in our model:

W_j : Cumulative On Duty Time at node j $\forall j \in N^P$

D_j : Uninterrupted Driving Time at node j $\forall j \in N^P$

A_j : Arrival Time at node j $\forall j \in N^P$

S_j : Short Break given at node j $\forall j \in N^P$

L_j : Long Break given at node j $\forall j \in N^P$

$$lb_j : \begin{cases} 1, & \text{if the break at node } j \text{ is a long break} \\ 0, & \text{otherwise} \end{cases} \quad \forall j \in N^P$$

$$sb_j : \begin{cases} 1, & \text{if there is a short break at node } j \\ 0, & \text{otherwise} \end{cases} \quad \forall j \in N^P$$

$$y_{tj} : \begin{cases} 1, & \text{is 1 if the arc emanating from } j, \\ & (j,k) \in A^P \text{ is entered at time } t \\ 0, & \text{otherwise} \end{cases} \quad t = 1..T_{MAX}, \forall j \in N^P$$

$zplus_{tj}$: Auxiliary variable to enforce feasibility $t = 1..T_{MAX}, \forall j \in N^P$

Decision variables W_j, D_j and A_j are required to keep track of cumulative time usage at each node. S_j and L_j are used to determine the length of the breaks at each node. We need binary decision variables lb_j and sb_j since at a specific node there can either be a short break or a long break.

2.5.1.2 Mixed Integer Programming Model

$(MIP-1)$

$$min \quad \sum_{i:i\in N^P} \sum_{j:(i,j)\in A^P} \sum_{t} y_{ti} R_{tij} \tag{2.1}$$

$$subject\ to$$

$$W_j \geq W_i + S_i - M lb_i + (1 - lb_i) d_{ij} \quad \forall (i,j) \in A^P \tag{2.2}$$

$$W_j \geq d_{ij} \quad \forall (i,j) \in A^P \tag{2.3}$$

$$W_j \leq W_{MAX} \quad \forall j \in N^P \tag{2.4}$$

$$D_j \geq D_i + d_{ij} - (lb_i + sb_i) M \quad \forall (i,j) \in A^P \tag{2.5}$$

$$D_j \geq d_{ij} \quad \forall (i,j) \in A^P \tag{2.6}$$

$$D_j \leq D_{MAX} \quad \forall j \in N^P \tag{2.7}$$

$$A_j = A_i + S_i + L_i + d_{ij} \quad \forall (i,j) \in A^P \tag{2.8}$$

$$A_d \leq T_{MAX} \tag{2.9}$$

$$S_j \leq S_{MAX} \quad \forall j \in N^P \tag{2.10}$$

$$S_j \leq M sb_j \quad \forall j \in N^P \tag{2.11}$$

$$L_j \geq L_{MIN} lb_j \quad \forall j \in N^P \tag{2.12}$$

$$L_j \leq M lb_j \quad \forall j \in N^P \tag{2.13}$$

$$lb_j \leq 1 - sb_j \quad \forall j \in N^P \tag{2.14}$$

$$D_j - D_{MAX} \leq lb_j - 1 \quad \forall j \in N^P \tag{2.15}$$

$$zplus_{tj} \leq (1 - y_{tj}) M \quad \forall t, j : j \in N^P \tag{2.16}$$

$$\sum_{t} y_{tj} = 1 \quad \forall j \in N^P \tag{2.17}$$

$$A_j + S_j + L_j = ty_{tj} + zplus_{tj} \quad \forall t, j : j \in N^P \tag{2.18}$$

$$A_j, W_j, D_j, S_j, L_j \quad is\ nonnegative\ integer \quad \forall j \in N^P \tag{2.19}$$

$$zplus_{tj} \quad is\ nonnegative\ integer \quad \forall t, j : j \in N^P \tag{2.20}$$

$$y_{tj} \in \{0,1\} \quad \forall t, j : j \in N^P \tag{2.21}$$

$$lb_j, sb_j \in \{0,1\} \quad \forall j \in N^P \tag{2.22}$$

Constraints (2.2), (2.5) and (2.3), (2.6) implies that the On Duty and Driving Time statistics are transferred from node i to j such that $(i,j) \in A^P$. Constraints (2.4) and (2.7) ensure that the On Duty and Driving Time restrictions are not broken. Constraint (2.8) imply that Arriving Time statistics are transferred from node i to j such that $(i,j) \in A^P$. Constraint (2.9) ensures that the node d

is reached within T_{MAX}. Constraints (2.10), (2.11), (2.12) and (2.13) are the lower and upper bound requirements for the Short and Long Breaks where M is a sufficiently large number. Constraint (2.14) implies that a short break and a long break cannot occur at the same node. Constraint (2.15) implements the driving time regulation which states that whenever uninterrupted driving time reaches D_{MAX} units, a long break must be given. Constraint (2.16) forces the slack variable $zplus_{tj}$ to be 0 if y_{tj} is 1. Constraint (2.17) implies that an arc can only be entered at on t. Constraint (2.18) ensures that random variable y_{tj} takes the value 1 only if the arc $(j,k) \in A^P$ is entered at time t such that $t = A_j + S_j + L_j$. Constraints (2.16), (2.17) and (2.18) together ensure that in the objective function, for each arc only the risks associated with the time that the truck enters the arc are considered.

This model is tested using CPLEX 11 Solver on four different paths with the parameters on Table 2.2. We set upper bound on the computation time of the solver as 3 hours since the Routing and Scheduling module requires many risk calculations in the algorithm. We define time periods as 5-minute intervals

Table 2.2: The Parameter Values Used in the Test Runs

Parameter	**Time-Periods**	**Real Time**
W_{MAX}	180 periods	15 hours
D_{MAX}	120 periods	10 hours
L_{MIN}	96 periods	8 hours
S_{MAX}	95 periods	7.9 hours
T_{MAX}	576 periods	48 hours

Table 2.3: Paths Used in MIP-1

Instance	**Path Length**
1	20 Nodes
2	20 Nodes
3	40 Nodes
4	40 Nodes

The runs of all of the instances terminate after 3 hours without any optimal solution. Formulation MIP-1 is not fast enough to be implemented effectively in

the algorithm. We simplify the model by excluding the driving time constraints (2.5), (2.6), (2.7), (2.15) and obtain a new formulation (MIP-2).

$(MIP-2)$

$$
\begin{aligned}
& min && \sum_{i:i\in N^P}\sum_{j:(i,j)\in A^P}\sum_{t} y_{ti}R_{tij} && \\
& subject\ to && && \\
& (2.2) && - \quad (2.4) && \\
& (2.8) && - \quad (2.14) && \\
& (2.16) && - \quad (2.18) && \\
& A_j, W_j, S_j, L_j && \quad is\ nonnegative\ integer && \forall j \in N^P \\
& zplus_{tj} && \quad is\ nonnegative\ integer && \forall t, j : j \in N^P \\
& y_{tj} && \in \{0,1\} && \forall t, j : j \in N^P \\
& lb_j, sb_j && \in \{0,1\} && \forall j \in N^P
\end{aligned}
$$

The drawback of this new formulation is that the feasible solutions of MIP-2 may not be feasible to MIP-1 in terms of driving time restrictions. We test this formulation on the same problem instances in Table 2.3 with the same parameters in Table 2.2.

MIP-2 does not provide sufficient improvement in the computation time, since each four problem instances terminate without any optimal solutions after 3 hours. Thus, we reformulate MIP-1 to obtain MIP-3 with the following changes: If the binary variable lb_j is 1 at node j, we know that there is a break of at least L_{MIN} periods. Then, if we exclude the constraints (2.11), (2.12), (2.13) and (2.14); a long break at node j can be represented as $lb_j L_{MIN} + S_j$. Then, we obtain a new formulation (MIP-3), when the decision variable L_j in constraints (2.8) and (2.18) is replaced with $lb_j L_{MIN}$.

$(MIP-3)$

$$
\begin{aligned}
& min && \sum_{i:i\in N^P}\sum_{j:(i,j)\in A^P}\sum_{t} y_{ti}R_{tij} && && \\
& subject\ to && && && \\
& (2.2) && - && (2.7) && \\
& A_j && = && A_i + S_i + lb_i L_{MIN} + d_{ij} && \forall (i,j) \in A^P && (2.23)\\
& (2.9) && - && (2.10) && \\
& (2.15) && - && (2.17) && \\
& A_j + S_j + lb_j L_{MIN} && = && ty_{tj} + zplus_{tj} && \forall t, j : j \in N^P && (2.24)\\
& A_j, W_j, S_j, D_j && && is\ nonnegative\ integer && \forall j \in N^P && \\
& zplus_{tj} && && is\ nonnegative\ integer && \forall t, j : j \in N^P && \\
& y_{tj} && \in && \{0,1\} && \forall t, j : j \in N^P && \\
& lb_j, sb_j && \in && \{0,1\} && \forall j \in N^P &&
\end{aligned}
$$

The only drawback of this approach is that we limit a long break by $L_{MIN} + S_{MAX}$ periods. The computational runs on the same problem instances using MIP-3 again terminate after 3 hours without any optimal solutions.

Since MIP-3 also does not provide any improvement, we combine the changes in MIP-2 and MIP-3 to obtain a new formulation (MIP-4), but the computational runs on the same problem instances using MIP-4 also do not provide optimal solutions after 3 hours.

$(MIP-4)$

$$
\begin{array}{rcll}
min & & \displaystyle\sum_{i:i\in N^P}\sum_{j:(i,j)\in A^P}\sum_{t} y_{ti}R_{tij} & \\
subject\ to & & & \\
(2.2) & - & (2.7) & \\
(2.9) & - & (2.10) & \\
(2.16) & - & (2.17) & \\
(2.23) & - & (2.24) & \\
A_j, W_j, S_j & & is\ nonnegative\ integer & \forall j \in N^P \\
zplus_{tj} & & is\ nonnegative\ integer & \forall t, j : j \in N^P \\
y_{tj} & \in & \{0,1\} & \forall t, j : j \in N^P \\
lb_j, sb_j & \in & \{0,1\} & \forall j \in N^P
\end{array}
$$

Long solution times of four formulations shows that Integer Programming formulations are not practical to be used as a Scheduling method.

2.5.2 Dynamic Programming Approach

We use the Dynamic Programming model that is proposed in Erkut and Alp [8]. This formulation fully represents our Scheduling Approach if the path we input is viewed as the network. Although, this approach yields an optimal solution around 30 seconds for a path which is considerably faster than the Integer Programming approach, it is not fast enough to calculate all possible paths in a neighborhood. Thus we decide to propose a heuristic approach. The computational complexity of the dynamic programming approach is $O(T_{MAX}D_{MAX}W_{MAX}(|N|) + T_{MAX}|N|logT_{MAX})$.

2.5.3 Heuristic Approach

A heuristic procedure is developed to construct schedules faster than Integer and Dynamic Programming Approaches, since the Route Selection and Neighborhood

Generation module requires many risk calculations throughout the algorithm. The algorithm we propose basically constructs a schedule by distributing breaks one by one to each node. There are mainly two stages in the algorithm. The first stage is implemented if the path is feasible without any long breaks. Starting with a schedule with no breaks, the marginal change that adding one more unit of break at each node is calculated and the break is assigned to the node that gives the lowest risk. The next iteration starts with the best solution of the previous one and again the change in the risk when one more unit of break is assigned to each node is calculated. This stage is terminated when the long break feasibility is broken and the algorithm moves to the second stage which is implemented to ensure the long break feasibility. In this stage, all possible combinations of allocating L_{MIN} amount of break to each node is searched. A combination of two breaks of size L_{MIN} can also be distributed if required. Afterwards, on a path where L_{MIN} amount of break is allocated to a given node, rest of the breaks are again distributed one by one to all nodes in the same manner. The schedule that is feasible and gives the lowest risk after all of these procedures is recorded. As mentioned before the amount of break to be distributed is calculated from $T_{MAX} - T_P$, where $T_P = \sum_{(i,j)\in A^P} d_{i,j}$ is the total traversing time of the path without any breaks. The pseudo-code of this procedure is given as Algorithm 3 and Algorithm 4.

The time complexity of this procedure is $|N| + T_{MAX}|N|^2 + |N|^2(|N| + T_{MAX}|N|^2)$ which is $O(T_{MAX}|N|^4)$. Distributing the breaks of size L_{MIN} increases the time complexity of the algorithm. However, this stage is required to ensure the long break feasibility. Suppose that stage 2 is not implemented and stage 1 terminates whenever the total duration reaches T_{MAX}. Then, given a path that requires at least one long break, none of the nodes is guaranteed to be assigned a break of more than L_{MIN} time units and the procedure may terminate without any feasible schedule.

The numerical tests we performed on this algorithm shows that it takes an average of 2.5 seconds to calculate a path of 30 nodes (P_1) and 13 seconds for a path of 40 nodes (P_2), so it is satisfactory in terms of obtaining a solution in a short period of time. Furthermore, allocating L_{MIN} amount of breaks take the

Algorithm 3 Scheduling Heuristic

Given a path $P'(N', A')$, initialize
for each node $i \in N'$ **do**
  Set $B_C[i] := 0$
  Set $B_I[i] := 0$
  Set $B_G[i] := 0$
end for
Set $R_G := M$
Calculate current risk R_C and amount of breaks to be distributed
while Maximum $W[i] \leq W_{MAX}, \forall i \in N'$ **do**
  Set $R_I := M$
  Set $B_C := B_I$
  for all $i \in N'$ **do**
    $B_C[i] := B_C[i] + 1$
    Calculate the current risk R_C
    if $R_C < R_I$ **then**
      Set $B_I := B_C$
      Set $R_I := R_C$
    end if
    if $R_C < R_G$ and the schedule is feasible **then**
      Set $B_G := B_C$
      Set $R_G := R_C$
    end if
    $B_C[i] := B_C[i] - 1$
  end for
end while
if Maximum $W[i] \geq W_{MAX}, \forall i \in N'$ and T_{MAX} not reached **then**
  for all $i \in N'$ **do**
    for all $j \in N'$ **do**
      for all $k \in N'$ **do**
        Set $B_C[k] := 0$
        Set $B_I[k] := 0$
      end for
      Set $B_C[i] = W_{MIN}$ and $B_C[j] = W_{MIN}$
      Calculate the number of remaining breaks to be distributed
      while T_{MAX} not reached **do**
        Set $R_I := M$
        Set $B_C := B_I$
        for all $k \in N'$ **do**
          $B_C[k] := B_C[k] + 1$
          Calculate the current risk R_C - Algorithm 4
          if $R_C < R_I$ **then**
            Set $B_I := B_C$
            Set $R_I := R_C$
          end if
          if $R_C < R_G$ and the schedule is feasible **then**
            Set $B_G := B_C$
            Set $R_G := R_C$
          end if
          $B_C[k] := B_C[k] - 1$
        end for
      end while
    end for
  end for
end if

Algorithm 4 Calculate Risk

Set $t := 0$, $v := 0$, $w := 0$
Set $risk := 0.0$
Set $maxt = 0$, $maxv = 0$, $maxw = 0$
for all arcs $(i, j) \in A^P$ in its order in P **do**
  $risk := risk + R_{ij(t+B_C[i])}$
  if $B_C[i] = 0$ **then**
    $v := v + d_{ij}$
    $w := w + d_{ij}$
  end if
  if $B_C[i] \leq S_{MAX}$ **then**
    $v := d_{ij}$
    $w := w + d_{ij}$
  end if
  if $B_C[i] \geq L_{MIN}$ **then**
    $v := d_{ij}$
    $w := d_{ij}$
  end if
  $t := t + d_{ij}$
  if $maxv < v$ **then**
    $maxv := v$
  end if
  if $maxw < w$ **then**
    $maxw := w$
  end if
end for
Return $risk$, $maxt$, $maxv$, $maxw$

majority of these CPU times, since a complete iteration of distributing breaks of size one to all nodes take 0.004 seconds for P_1 and 0.019 seconds for P_2.

The risks of the schedules are not guaranteed to be the optimal because of the heuristic nature of this approach. But we realize that it terminates with risks closer to optimal, when the total amount of breaks to be distributed is relatively small. The reason of this situation is that, the Scheduling Heuristic is greedy in the sense that at each iteration once the position of a break is fixed it does not change until the end of that iteration. For instance, a solution with one unit of a break at node $k \in N^P$ may be the optimal when $T_{MAX} = T_P + 1$, on the other hand, when $T_{MAX} = T_P + 2$, the optimal solution may not have any breaks assigned to node k. But, since the breaks are distributed one by one and fixed, previously assigned breaks create a bias in the next assignment. Then, this bias increases when $T_{MAX} - T_P$ increases. The algorithm also find risks closer to optimal, when all of the short break feasibility requirements are relaxed. That is, the algorithm allows solutions that are infeasible due to short break requirements be the best solution of an iteration hoping to achieve short break feasibility at the next iterations. Moreover, the short break feasibility is not checked when the path is recorded as the best solution of the algorithm and the best solution of iteration. The usage of this approach may seem to cause problems in the main heuristic because the route selection decisions are based on the risks obtained from this module. However, in the intermediate iterations of the main heuristic, we only need to rank the paths in terms of risks. There is not any necessity to obtain the optimal risks. Since it is not guaranteed, if the ranking is done correctly, the main heuristic will eventually reach the optimal path without actually requiring the optimal risk. When the main heuristic terminates, the optimal risk of the path can be calculated using the Dynamic Programming Approach.

2.6 Termination Criteria

The myopic property of Scheduling and Risk Calculation module may cause ranking problems at the Route Selection and Neighborhood Generation module to select paths with higher optimal risks instead of paths with lower optimal risks. Then, the Main Heuristic procedure may discard or get away from the optimal path. Thus, we need to keep record of the most promising paths throughout the algorithm in SBP_I and SBP_A, since, using the Dynamic Programming Approach, we can calculate the optimal risks of these paths and select the best one. When the S_{SBP_I} is fixed, then the Main Heuristic terminates whenever a BP_I of an iteration could not enter SBP_I, that means, there will be no new paths in SBP_I to generate neighborhood.

Chapter 3

Numerical Experiments

In this chapter, we present our experimental results that we obtain using the methodology described in Chapter 2 and we compare them with the solutions obtained using the dynamic programming formulation proposed by Erkut and Alp [8].

3.1 Parameter Settings

Our methodology requires mainly two different types of parameters to be determined: Time-related parameters and algorithm-related parameters. The time-related parameters are the driving (D_{MAX}), on duty (W_{MAX}) and the maximum trip time (T_{MAX}) restrictions. The algorithm-related parameters are sizes for Set of Best Paths of Neighborhood (S_{SBP_N}), Set of Best Paths of Iteration (S_{SBP_I}) and Set of Best Paths of Algorithm (S_{SBP_A}) and the settings for the neighborhood contraction methods which are Window Shifting (WS) and Cumulative Occurrence (CO). In window shifting, window size parameter w, in cumulative occurrence, percent parameter λ and warm-up parameter ϵ should be determined. We test our problems mainly for two different maximum trip time conditions and several different algorithm-related parameter settings given in Table 3.1. We discretized the time into 5-minute intervals in our input data and set the total

available time to 360 and 576; hence 360 periods in the algorithm correspond to 1800 minutes in real time and 576 periods in the algorithm corresponds to 2880 minutes in real time. We implement the algorithm and run all test problems with no neighborhood contraction (NC) and with WS and CO methods. In WS, we set the window size w to 3 or 5. In CO, we set the percent parameter λ from 10 to 70 with a step of 10 and the warm-up period parameter ϵ to 1, 2, or 3 iterations.

Table 3.1: The Parameter Settings Used In Test Networks

Parameter	**Values**
T_{MAX}	360, 576 periods
D_{MAX}	120 periods
W_{MAX}	180 periods
w	3, 5 iterations
λ	10, 20, ... , 70%
ϵ	1, 2, 3 iterations
S_{SBP_N}	10
S_{SBP_I}	10
S_{SBP_A}	10

We implement our heuristic with the above mentioned parameters and we run the DP algorithm for each path in the Set of Best Paths of Algorithm and the Set of Best Paths of Iteration after the heuristic terminates.

3.2 Test Networks

Our first test network is the network that is used by Nozick et. al. [15] and Erkut and Alp [8] in their numerical experiments. We refer this network as NLT. This network represents a portion of northeast USA and it is composed of 138 nodes and 368 arcs, where durations and traversing risks associated with each arc is provided for a 24-hour period.

We also consider seven different sample networks of various sizes. We used test networks for Steiner Tree problems that are available through the internet as a basis to generate our sample networks. The sizes of these networks are given

Table 3.2: The Sample Network Properties

Network	Number of Nodes	Number of Arcs
Network 1	100	248
Network 2	100	398
Network 3	75	295
Network 4	100	248
Network 5	100	248
Network 6	100	397
Network 7	100	397

in Table 3.2. Since these test networks do not contain any population or risk information, we generated risk information based on NLT. When the unit risk attributes of NLT (risks per unit distance) are obtained for each arc, it can be seen that there are mainly two different unit risk patterns for a 24-hour period; one for populated areas and one for unpopulated areas. So, we can obtain a risk pattern for each arc on the sample networks using their arc lengths. We randomly determine the population information of the arcs with 60% probability of being unpopulated and 40% being populated.

3.3 Discussion

We take three origin - destination pairs on Network NLT (denoted by NLT 1, NLT 2, NLT 3) and one origin - destination pair for each Sample Network, making a total of 10 test networks, and implement our methodology on a 1500 MHz dual CPU running Solaris Operating System. The results of all networks with all parameter combinations are given in Appendix C. The results that we obtain from the computational tests for one parameter setting for each network for NC, WS and CO cases are summarized in Tables 3.3, 3.4 and 3.5, respectively.

In Tables 3.3, 3.4 and 3.5, the column "Minimum Heuristic Risk" represents the minimum risk that our risk calculation algorithm finds. The "Optimal Risk of Heuristic" column in this table represents the minimum risk that is obtained after solving each path in SBP_I and SBP_A with the DP algorithm. The values

Table 3.3: The Computation Results without Neighborhood Contraction Methods

Network	T_{MAX}	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
NLT 1	360	15.201	66,639.048	65,730	65,730	0
NLT 2	360	6.893	18,567.486	18,567.486	18,567.486	0
NLT 3*	360	-	-	-	37,915	-
Network 1	360	18.336	8,948	8,948	8,948	0
Network 2	360	145.463	6,653.466	3,789.85	3,789.85	0
Network 3	360	84.383	7,866	4,871.62	2,765.01	76.18
Network 4	360	7.755	2,182.261	1,897.52	1,897.52	0
Network 5	360	21.618	18,981	18,981	18,981	0
Network 6	360	178.0191	1,629.186	774.628	774.628	0
Network 7	360	37.125	3,067.502	779.631	779.631	0
NLT 1	576	424.470	36,590.715	27,613	27,613	0
NLT 2	576	119.709	18,567.486	7,189	7,189	0
NLT 3	576	36.579	22,510.912	17,244	17,244	0
Network 1	576	82.007	6,143.775	4,692	2,260.44	107
Network 2	576	553.509	6,653.466	1,909.84	1,909.84	0
Network 3	576	337.983	7,866	2,574	1,662.59	54.81
Network 4	576	33.225	2,182.261	1,897.52	1,897.52	0
Network 5	576	95.558	11,858.886	5,412	5,240	3.28
Network 6	576	691.567	1,629.186	774.628	774.628	0
Network 7	576	127.322	3,067.502	700.439	700.439	0

*: The heuristic algorithm could not find any feasible solution for $T_{MAX} = 360$ case, since the minimum feasible trip time is 355 periods which is very close to 360.

in the "Optimal Risk" column are the risks that are obtained by solving the complete network by the DP algorithm (this is the global optimal to the original integrated routing and scheduling problem). The "Gap" column represents the percentage gap between the optimal risk of heuristic and the optimal risk.

In our computational tests, the performance criteria that we use to compare the results is whether the optimal path resides in the sets SBP_A or SBP_I. Note that, if this is the case, then we are able to find the global optimal to the integrated problem as we are proposing to implement the DP algorithm on the paths that appear in these two sets. According to the results we obtained for a total of ten different networks, we summarize the performance of our heuristic as follows:

- In the NC and WS cases, our algorithm finds the optimal paths in eight

Table 3.4: The Computation Results using Window Shifting with $w = 5$ for $T_{MAX} = 360$ and $w = 3$ for $T_{MAX} = 576$

Network	T_{MAX}	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
NLT 1	360	22.293	66,639.048	65,730	65,730	0
NLT 2	360	5.498	18,567.486	18,567.486	18,567.486	0
NLT 3*	360	-	-	-	37,915	-
Network 1	360	10.997	8,948	8,948	8,948	0
Network 2	360	128.400	6,653.466	3,789.85	3,789.85	0
Network 3	360	52.181	7,866	4,871.62	2,765.01	76.18
Network 4	360	4.925	2,182.261	1,897.52	1,897.52	0
Network 5	360	15.106	18,981	18,981	18,981	0
Network 6	360	109.237	1,629.186	774.628	774.628	0
Network 7	360	67.169	3,067.502	779.631	779.631	0
NLT 1	576	359.787	36,590.715	27,613	27,613	0
NLT 2	576	19.270	18,567.486	7,189	7,189	0
NLT 3	576	24.126	22,510.912	17,244	17,244	0
Network 1	576	65.767	6,143.775	4,692	2,260.44	107
Network 2	576	327.452	6,653.466	1,909.84	1,909.84	0
Network 3	576	1.120	7,866	2,574	1,662.59	54.81
Network 4	576	3.819	2,182.261	1,897.52	1,897.52	0
Network 5	576	110.348	11,858.886	5,412	5,240	3.28
Network 6	576	371.118	1,629.186	774.628	774.628	0
Network 7	576	539.413	3,067.502	700.439	700.439	0

*: The heuristic algorithm could not find any feasible solution for $T_{MAX} = 360$ case, since the minimum feasible trip time is 355 periods which is very close to 360.

networks for $T_{MAX} = 360$ case and seven networks for $T_{MAX} = 576$ case. When CO method is implemented, our algorithm finds the optimal paths in seven networks for both $T_{MAX} = 360$ and $T_{MAX} = 576$ cases.

- The risk calculation heuristic component of our algorithm finds the optimal risks in three networks for $T_{MAX} = 360$ case in each of the NC, WS and CO cases. The risk calculation heuristic does not find any optimal risk for $T_{MAX} = 576$.

These results imply that in each of the NC, WS and CO cases, the risk calculation heuristic component of our algorithm does not find the optimal risks in seven networks for $T_{MAX} = 360$. However, in five of these seven networks the optimal path is found in sets SBP_A or SBP_I for NC and WS cases whereas this

Table 3.5: The Computation Results using Cumulative Occurrence Method with $\lambda = 30$ and $\epsilon = 3$

Network	T_{MAX}	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
NLT 1	360	4.650	66,639.048	65,808.9	65,730	0.12
NLT 2	360	1.404	18,567.486	18,567.486	18,567.486	0
NLT 3*	360	-	-	-	37,915	-
Network 1	360	9.489	8,948	8,948	8,948	0
Network 2	360	24.517	6,653.466	3,789.85	3,789.85	0
Network 3	360	0.427	7,866	4,871.62	2,765.01	76.18
Network 4	360	1.232	2,182.261	1,897.52	1,897.52	0
Network 5	360	5.425	18,981	18,981	18,981	0
Network 6	360	14.489	1,629.186	774.628	774.628	0
Network 7	360	0.189	3,067.502	779.631	779.631	0
NLT 1	576	70.752	36,590.715	27,613	27,613	0
NLT 2	576	19.135	18,567.486	7,189	7,189	0
NLT 3	576	16.918	22,510.912	17,244	17,244	0
Network 1	576	40.184	6,143.775	4,692	2,260.44	107
Network 2	576	93.495	6,653.466	1,909.84	1,909.84	0
Network 3	576	1.344	7,866	2,574	1,662.59	54.81
Network 4	576	4.799	2,182.261	1,897.52	1,897.52	0
Network 5	576	24.309	11,858.886	5,412	5,240	3.28
Network 6	576	54.536	1,629.186	774.628	774.628	0
Network 7	576	0.378	3,067.502	700.439	700.439	0

*: The heuristic algorithm could not find any feasible solution for $T_{MAX} = 360$ case, since the minimum feasible trip time is 355 periods which is very close to 360.

number is four for CO case. Similarly, when $T_{MAX} = 576$, the risk calculation heuristic does not find the optimal risk in ten networks and in eight of these ten networks the optimal path is found in sets SBP_A or SBP_I for CO, NC and WS cases. This situation brings out the importance of using the DP algorithm since the heuristic risk calculation may output a wrong schedule on the optimal path. The gaps associated with these cases are also zero since the optimal risks are achieved.

Furthermore, when we examine the results with nonzero gaps, in the NC and WS cases, we do not obtain the optimal risk in one network for $T_{MAX} = 360$ and two networks for $T_{MAX} = 576$, because the optimal paths are not found in sets SBP_A or SBP_I. Similarly, in CO case, we do not obtain the optimal risk in two

networks for $T_{MAX} = 360$ and two networks for $T_{MAX} = 576$. Solving the paths in these sets optimally with the DP algorithm is again beneficial for these paths since the risk calculation component of our heuristic algorithm may build a wrong schedule over a wrong path. Especially in Network 3, the percentage gap between the columns "Optimal Risk of Heuristic" and "Optimal Risk" is 54.81%, while the gap between columns "Minimum Risk of Heuristic" and "Optimal Risk" is around 300%.

According to the computational results given in Tables 3.3, 3.4 and 3.5, there is a gap between the optimal risk and the optimal risk of heuristic in some of the networks. This gap is as high as 107%. The possible reason for this situation is the heuristic and the greedy nature of the risk calculation module. For instance in Network 1 for $T_{MAX} = 576$ there is a 107% gap between the optimal solution and the heuristic solution for the CO with $\lambda = 30$ and $\epsilon = 3$ case. However, for the CO with $\lambda = 30$ and $\epsilon = 1$ case (see Appendix C for details), the heuristic finds the optimal path. When we examine the risks obtained by paths in the Set of Best Risk of Algorithm for each case, we observe that our heuristic algorithm calculates the risk of the optimal path of Network 1 for $T_{MAX} = 576$ as 8948.432, which is in fact 2260.440 when calculated by DP algorithm. Moreover, there exists other paths that the heuristic risk calculation module calculates to have smaller risk values (Table 3.6).

We can see from Table 3.6 that the Set of Best Paths of Algorithm for CO $\lambda = 30$ and $\epsilon = 3$ case is composed of risks smaller than 8948.432 and the path with the minimum optimal risk among the set is calculated to be 4692 which causes the 107% gap. When we run our algorithm using CO ($\lambda = 30$, $\epsilon = 3$) with set sizes $S_{SBP_A} = 40$ and $S_{SBP_I} = 40$, we observe that the algorithm terminates with the optimal path ranked 33^{rd} in the Set of Best Paths of Algorithm. This implies that our heuristic procedure actually constructs the optimal path at an iteration but due to the risk calculation module, the ranking of the paths is not done correctly, thus the optimal path is discarded before the algorithm terminates.

The heuristic risk calculation module performs better for smaller T_{MAX} values. Especially, the optimal risks are found only for $T_{MAX} = 360$ case of three

Table 3.6: The Comparison of Heuristic Risks of Paths in the Set of Best Paths of Algorithm of Network 1

	Heuristic Risk	
	CO $\lambda = 30$ **and** $\epsilon = 3$	**CO** $\lambda = 30$ **and** $\epsilon = 1$
1	6143.775	6143.775
2	**6602.455**	7031.408
3	7031.408	7051.646
4	7051.646	7320.659
5	7320.659	8935.350
6	7552.199	**8948.432**
7	7582	9716.379
8	7652.448	9990.071
9	7717.968	10336.511
10	7886.540	10578.190
Optimal Risk	**4692**	**2260.440**

networks. The greedy property of the risk calculation module is the reason for this situation since the increase in the amount of breaks to be distributed, $T_{MAX} - T_P$, may deviate the search away from the optimal schedule as discussed in Chapter 2.

Even though there exists a gap between the risks of optimal solution and heuristic solution, the realization of the paths do not significantly differ from the optimal one. For instance, the optimal path for the Network NLT 1 for $T_{MAX} = 576$ is given in Figure 3.1. The path that our heuristic finds for the same problem using CO method with $\lambda = 30$ and $\epsilon = 1$ has a 3.2% gap (See Appendix C). However, this path is very close to the optimal path as seen on Figure 3.1. Likewise, for the Network NLT 3 for $T_{MAX} = 576$, CO method with $\lambda = 30$ and $\epsilon = 1$ has a 17.03% gap from the optimal path, but the paths are nearly similar as given in Figure 3.2.

The usage of the neighborhood contraction methods and the settings of the algorithm related parameters significantly affect the CPU Time of the algorithm. For instance, the usage of neighborhood contraction methods decreases the computation time of NLT 2 for $T_{MAX} = 576$ case given at Tables 3.3 and 3.5 from 119.70 minutes to 19 minutes without losing the optimal solution. Furthermore,

Table 3.7: Detailed CPU Time Results of Network 5 with $\epsilon = 3$ for $T_{MAX} = 576$ for Different Values of λ

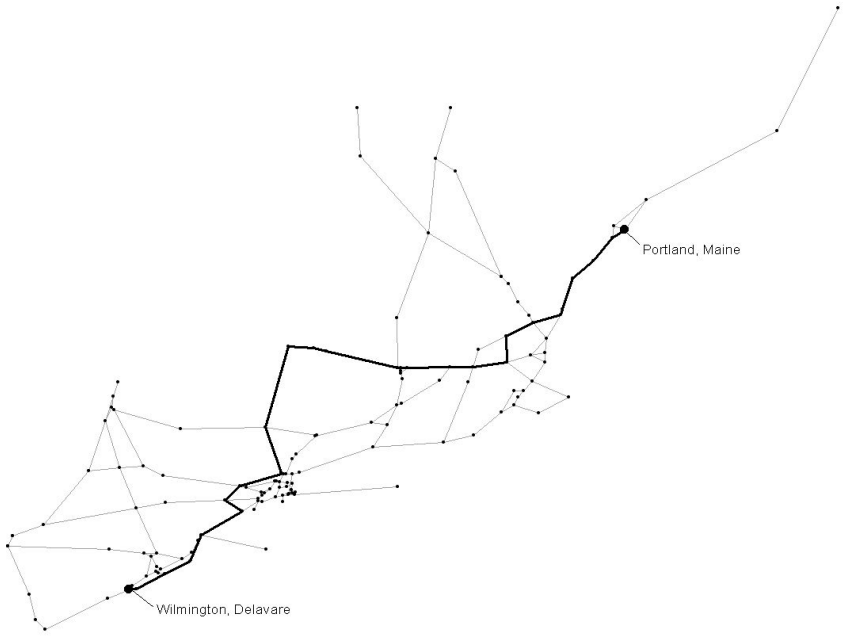
Percent Parameter λ	**CPU Time**	**Gap**
10	55.307	3.28
20	31.595	3.28
30	24.309	3.28
40	20.073	3.28
50	19.190	3.28
60	18.514	3.28
70	18.514	3.28

usage of WS instead of NC decreases the average CPU Time by 19% and 27% for $T_{MAX} = 360$ and $T_{MAX} = 576$ cases, respectively and usage of CO instead of NC decreases the average CPU Time by 87% and 86% for the same cases, respectively. Although we cannot conclude whether the WS or CO method is better in terms of risk and path outputs, CO method is better in terms of CPU time. According to the results given in Tables 3.4 and 3.5, usage of CO method instead of WS method decreases the average CPU time by 85% and 82% for $T_{MAX} = 360$ and $T_{MAX} = 576$ cases, respectively. Moreover, the increase in the percent parameter λ decreases the computation time, but there is not any significant improvement for $\lambda \geq 40$ as seen on Table 3.7. The decrease in the warm-up period ϵ which is required to collect statistics for the CO method also decreases the CPU time. In seven networks, decreasing the warm-up period to $\epsilon = 1$ results the same path output but with smaller CPU Time (see Appendix C for details).

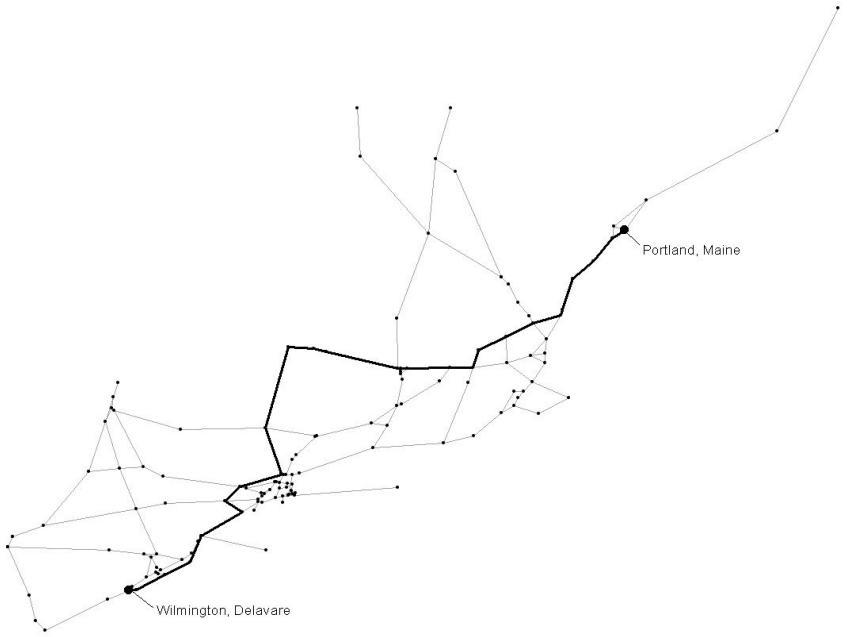
The set sizes S_{SBP_N}, S_{SBP_I}, S_{SBP_A} have different effects on the algorithm. If S_{SBP_I} is increased, more paths are allowed to be in the set SBP_I, thus it may take more time for the algorithm to reach its termination criteria. (Recall that, the algorithm terminates whenever the BP_I of an iteration could not enter SBP_I). If S_{SBP_A} is increased, then there will be more candidate paths in the set to be solved by DP algorithm when the algorithm terminates. The increase in the value of S_{SBP_N} changes the statistics collected for the Cumulative Occurrence method. Allowing more paths in SBP_N increases the appearance statistics of nodes which may cause more nodes to be significantly appearing. Then the neighborhood size for Cumulative Occurrence method may increase resulting a higher CPU Time.

One other important characteristic of our heuristic is that it is a more CPU demanding procedure than the DP method. The given network and its parameters have significant effects on the CPU time. The increase in the density of the network increases the neighborhood size, thus increase the total CPU time. For instance, the CPU time of Networks 4 and 5 which have 100 nodes and 248 arcs, are lower than that of Networks 6 and 7 which have 100 nodes and 397 arcs.

Regarding the computational results, we suggest using the Cumulative Occurrence method with $\lambda = 30$ and $\epsilon = 3$ since this parameter setting works in the majority of the cases with comparably smaller CPU times. The set size settings $S_{SBP_N} = 10$, $S_{SBP_I} = 10$, $S_{SBP_A} = 10$ are sufficient for most cases. So, we use these parameter settings for the computational test on large networks which is introduced in Chapter 4.

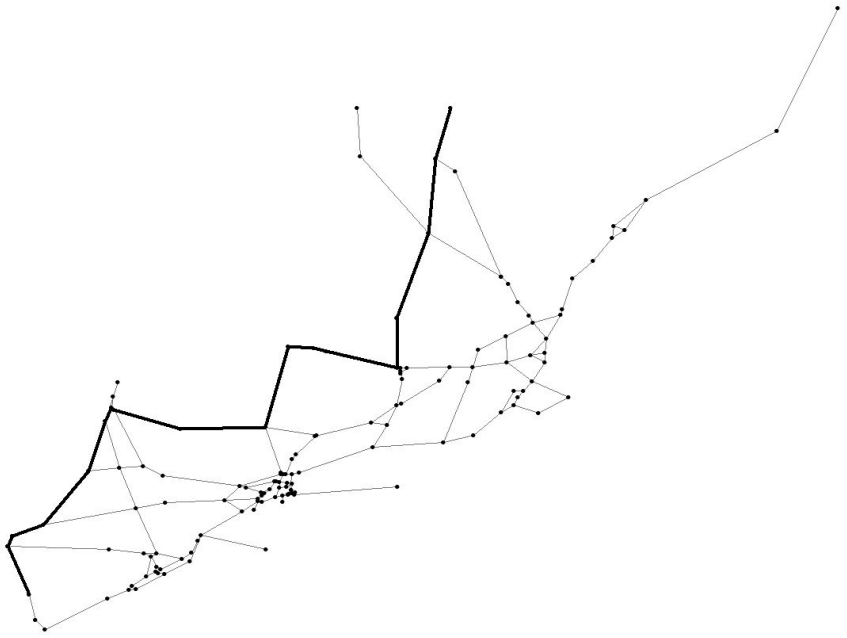


(a)

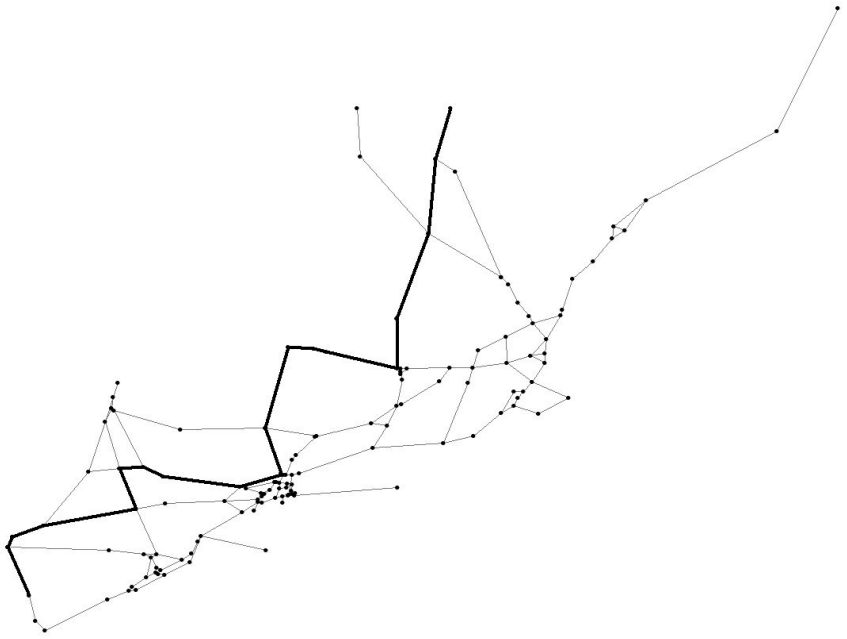


(b)

Figure 3.1: Network NLT 1 (a): The Optimal Path When $T_{MAX} = 576$, (b): The Path Obtained Using Cumulative Occurrence Method with $\lambda = 30$ and $\epsilon = 1$



(a)



(b)

Figure 3.2: Network NLT 3 (a): The Optimal Path When $T_{MAX} = 576$, (b): The Path Obtained Using Cumulative Occurrence Method with $\lambda = 30$ and $\epsilon = 1$

Chapter 4

Application on Large Networks

Our primary aim of developing the heuristic procedure explained in Chapter 2 is to implement our problem to larger sized networks, since the memory consumption of DP Algorithm increases as network gets larger. In Chapter 3, the networks are small enough to be solved by the DP Algorithm for comparison purposes. However, in this chapter, we present the implementation of our heuristic procedure on Turkey Road Network for transportation between major cities, border gates and harbors.

4.1 Characteristics of Turkey Road Network

According to the General Directorate of Highways of Turkey [22], the Turkey Road Network is 64033 kilometers long and mainly composed of three different types of roads:

- **Highways** (Otoyol): This type of roads allow high traveling speeds and they are available between major cities like İstanbul, Ankara, Edirne and Adana. These roads also act as a peripheral road around these cities which allow transit passage at any hour. The access of pedestrians or vehicles without motor to these roads is prohibited. According to the 2009 data,

the length of highways in Turkey is 2010 km.

- **State Roads** (Devlet Yolu): This type of roads connect each city and major towns all around Turkey. These roads pass through urban and rural areas. According to the 2009 data, the length of state roads in Turkey is 31311 km.

- **District Roads** (İl Yolu): This type of roads connect the other towns and villages. These roads also pass through urban and rural areas. According to the 2009 data, the length of district roads in Turkey is 30712 km.

According to the laws, the following speed limitations are applied to the trucks carrying hazardous materials in Turkey [22]:

Table 4.1: The Speed Limitations on Turkey Road Network for Vehicles Carrying Hazardous Materials

Road Type	**Speed Limit**
Highways (Urban and Rural Areas)	60 km/h
State and District Roads (Rural Areas)	50 km/h
State and District Roads (Urban Areas)	30 km/h

We consider these regulations in the calculations of the arc related parameters of the Turkey Road Network discussed in the next section.

4.2 Network Simplification and Parameter Generation

We have Turkey Road Network data for ESRI ArcView GIS 3.2 [23] which is a commonly used geographical information software. The raw network data requires some modifications in order to reflect our objectives and problem requirements.

The Turkey Road Network data contain arcs which have coordinate, code,

explanation and type attributes. There is also coordinate and population information of all cities and towns on the network. First of all, we need to create nodes on the intersection points of these arcs and obtain node-arc representation of the network to be fed into the heuristic algorithm. For this purpose, we use Point and Polyline (P/PL) Tools v1.2 for ArcView 3.2 which is a script toolbox available on the internet [21]. Afterwards, using the arc information about the road types, we eliminate the District Roads from the network because coverage of Highways and State Roads fulfill our objectives. The elimination of the District Roads results a network with 11078 arcs and many nodes with degree of two arcs. However, these nodes are no longer required since they originally belong to the intersection of District Roads and State Roads. So, in order to remove these nodes and further simplify the network, we concatenate the arcs emanating from these nodes using the P/PL Tools while preserving the required arc attributes like coordinate, road type and road number information. We also calculate the arc lengths of the final network using Path with Distances and Bearings tool for ArcView [26]. As a result, we obtain the final network given in Figure 4.1 which contains 1606 arcs connecting 571 nodes.

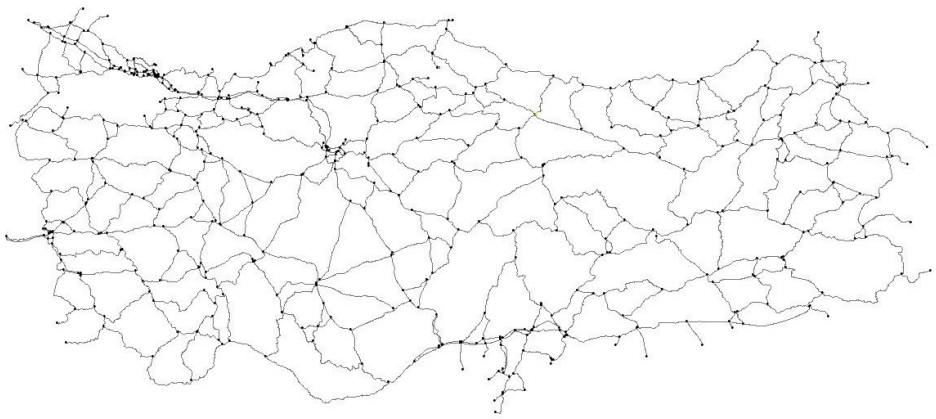


Figure 4.1: The Turkey Road Network After Modifications

In order to generate time-dependent risk attributes, we need to determine the number of people that are affected for each arc in the network. We use the methodology explained in Erkut and Verter [10] which is illustrated in Figure 4.2. For each arc in the network we calculate the population within a circle with radius

of r kilometers centered on each point on the arc using the Nearest Features v3.8b tool for ArcView [25]. We take $r = 1$ kilometers in our calculations. When we sum up all of population for an arc, we divide it by the length of the arc to get the population per kilometer.

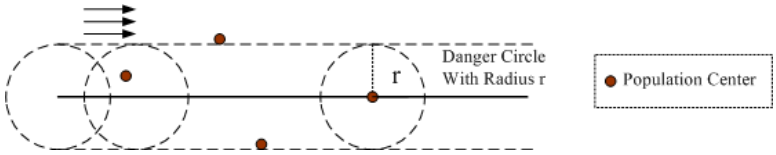


Figure 4.2: Illustration of Impact Radius

Table 4.2: Accident Release Probabilities per Million Vehicle-km

	Day (7 a.m. - 6 p.m.)	**Night (6 p.m. - 7 a.m.)**
Urban Roads	0.065	0.104
Rural Roads	0.028	0.044

We use the risk probabilities for different hours of the day given in Table 4.2 which is used by Erkut and Alp [8]. For urban roads, we also consider a detailed time slot structure given in Table 4.3 to reflect the effects of rush hour traffic as discussed in Chapter 1. We take $\alpha_1 = 2$ and $\alpha_2 = 1.5$ in our calculations. We obtain the time-dependent arc attributes by multiplying the probabilities by the population per kilometer values of each arc.

Table 4.3: Detailed Accident Release Probabilities for Urban Roads per Million Vehicle-km ($\alpha_1 > \alpha_2 > 1$)

Time Slot	**Probability**
7 a.m. - 10 a.m.	$0.065\alpha_1$
10 a.m. - 4 p.m.	0.065
4 p.m. - 6 p.m.	$0.065\alpha_1$
6 p.m. - 7 p.m.	$0.104\alpha_1$
7 p.m. - 9 p.m.	$0.104\alpha_2$
9 p.m. - 7 a.m.	0.104

In order to calculate the traversing times for each arc, we use the road type and arc length information of the arcs regarding the speed limitations given in Table 4.1. For the state roads, we assume that arcs whose total population attribute is greater than 100,000 belong to an urban area.

4.3 Computational Experiments

We take three source and destination points on Turkey Network namely, Kocaeli - Hakkari, Kocaeli - Adana and İzmir - Van, and implement out algorithm using Cumulative Occurrence method with the parameter settings given in Table 4.4.

Table 4.4: The Parameters Used in the Turkey Road Network Implementation

Parameter	**Explanation**	**Value**
T_{MAX}	Maximum Trip Time	576 periods
D_{MAX}	Maximum Uninterrupted Driving Time	120 periods
W_{MAX}	Maximum On Duty Time	180 periods
λ	Percent Parameter	30%
ϵ	Warm-up Parameter	3
S_{SBP_N}	Size of Set of Best Paths of Neighborhood	10
S_{SBP_I}	Size of Set of Best Paths of Iteration	10
S_{SBP_A}	Size of Set of Best Paths of Algorithm	10

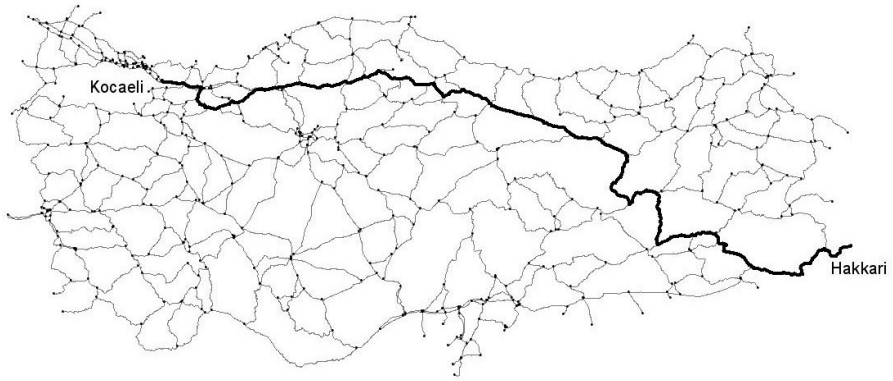
We solve the paths in the Set of Best Paths of Algorithm and the Set of Best Paths of Iterations by the DP Algorithm and report the risks associated with the path that yields the minimum among all in Table 4.5.

Table 4.5: The Computation Results of Turkey Road Network

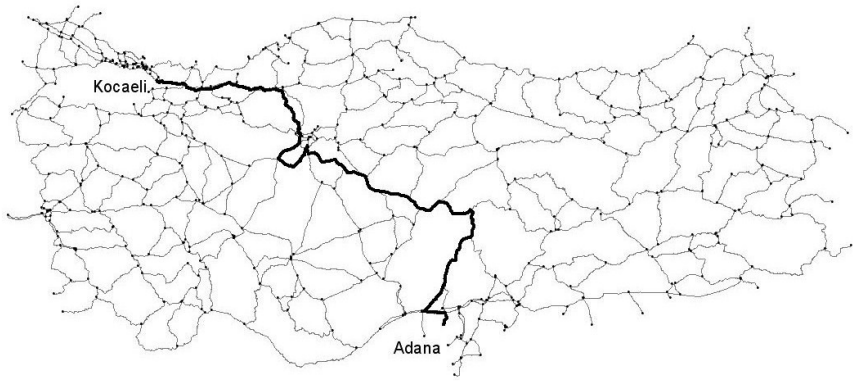
Origin, Destination Pair	T_{MAX}	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**
Kocaeli - Hakkari	700*	87.48	12529.9	12529.9
Kocaeli - Adana	576	265.43	11480.43	9168.56
İzmir - Van	700*	20.12	33946.34	33676.1

*: There is not any feasible path exist for these origin-destination pairs for $T_{MAX} = 576$ periods. The algorithm finds a feasible path for $T_{MAX} \geq 700$ periods.

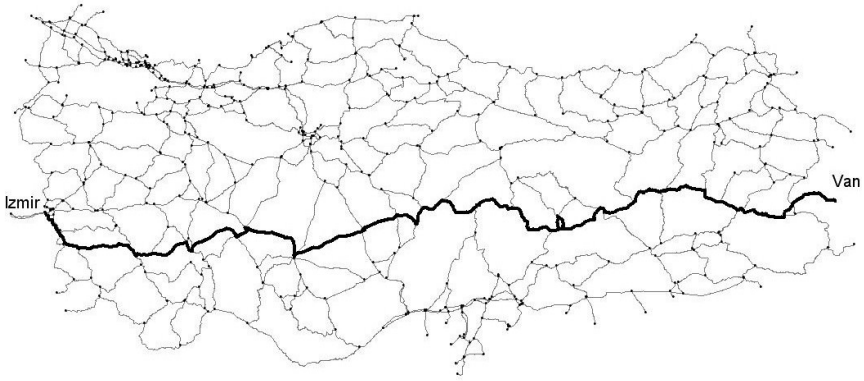
We illustrate the paths that are yield the minimum risk for each origin destination point in Figure 4.3.



(a)



(b)



(c)

Figure 4.3: (a): The Minimum Risk Path from Kocaeli to Hakkari, (b): The Minimum Risk Path from Kocaeli to Adana, (c): The Minimum Risk Path from İzmir to Van

Chapter 5

Conclusion

In this thesis, we consider an integrated routing and scheduling problem which arises in the context of transportation of hazardous materials. The problem is to find a route for a truck carrying hazardous materials between a specified origin and destination and to build a schedule to determine where and how long to stop on this route. The objective of the problem is to minimize the time dependent risk which is defined as the expected number of people affected in the presence of a possible accident. Although there is an exact formulation available in the literature for this problem, it is incapable of handling large networks due to the computational memory requirements. Thus, our aim is to develop a heuristic procedure that can solve the problem on larger networks.

We decompose the problem into two distinct operations, namely, routing and scheduling, and handle each component separately. This simplifies the scheduling problem because we find a driving schedule for a given single path on the network. Routing component is handled through a neighborhood search mechanism. We propose a neighborhood definition to generate several paths in the vicinity of a given path. This neighborhood structure is a generic one that it can be used in other network related problems as well. The main algorithm is an improvement heuristic based on a search mechanism using this proposed neighborhood. The procedure starts with an initial path and try to improve this path by moving to better paths in the neighborhood. We develop two neighborhood contraction

methods which use the statistical occurrence of the nodes throughout the algorithm in order to shrink the size of the neighborhoods, since the neighborhood function we define generates some paths that are not likely to be a candidate for the optimal path.

Given a path, scheduling component finds a driving schedule to minimize the total risk imposed during the trip. We first discuss that this problem is a resource allocation problem with time dependent attributes. We propose three different approaches to handle the problem. Firstly, we propose a Mixed Integer Programming formulation. We give four variations of the formulation, but since the time required to solve each of these formulations take large amount of time, we prefer not to use this approach. Secondly, we discuss using the Dynamic Programming Approach proposed by Erkut and Alp [8] but the improvement in the computation time is not sufficient enough to be implemented in our heuristic procedure. Hence, we develop a heuristic procedure for scheduling module and used it in our computations. However, since it is a heuristic approach, the risks calculated may differ from the optimal risk. Thus, we record the most promising paths generated throughout the algorithm and to solve these paths optimally using the DP algorithm.

The computational tests that we perform using our heuristic procedure show that the algorithm find the optimal path in the majority of the test networks rather than the optimal risk. This underlines the necessity of using the DP algorithm to solve the most promising paths optimally. Furthermore, when the cases in which our heuristic procedure do not find the optimal path are examined, it appears that the heuristic nature of the scheduling and risk calculation module causes this situation. If the optimal risks are calculated for each path throughout the algorithm (which takes considerable amount of time with the DP Algorithm), then the neighborhood search is more likely to find the optimal path. Moreover, if a better methodology is developed for the risk calculation, due to the modular structure of our heuristic algorithm, it can easily replace the existing risk calculation procedure without changing the whole algorithm.

Finally, we present the performance of our algorithm for different source and

destinations on Turkey Road Network which is a large sized network and show the best paths that are obtained for each different case.

Bibliography

[1] Ahuja, R. K. , T. L. Magnanti, J. B. Orlin. Network Flows *Prentice Hall*, 1993.

[2] Akgun V., A. Parekh, R. Batta and C. M. Rump. Routing of Hazmat Truck in the Presence of Weather Systems. *Computers and Operations Research*. **34** pp 1351-1373, 2007.

[3] Aneja Y. P. and K. P. K. Nair. The Constrained Shortest Path Problem. *Naval Research Logistics* **25**. pp. 549-555, 1978.

[4] Cai, X., T, Kloks and C. K. Wong. Time-Varying Shortest Path Problems with Constraints. *Networks*. **29**. pp. 141-149, 1997.

[5] Cooke, K. L. and E. Halsey. The Shortest Route Through a Network with Time Dependent Internodal Transit Times. *Journal of Mathematical Analysis and Applications*. **14**. pp.493-498, 1966.

[6] Cox R. G. Routing and Scheduling of Hazardous Materials Shipments: Algorithmic Approaches to Managing Spent Nuclear Fuel Transport. Phd. Dissertation. Cornell University, Ithaca, NY, 1984.

[7] Dumitrescu I., N. Boland. Improved Preprocessing, Labeling and Scaling Algorithms for the Weigth-Constrained Shortest Path Problem. *Networks*. **42** (3). pp. 135-153, 2003.

[8] Erkut E. and O. Alp. Integrated Routing and Scheduling of Hazmat Trucks with Stops En Route. *Transportation Science* **41**. pp. 107-122, 2007.

[9] Erkut E and V. Verter. Hazardous Materials Logistics *Facility Location: A Survey of Applications and Methods*. Editor: Zvi Drezner, **20**. pp. 467-506. 1995.

[10] Erkut, E. and V. Verter. Modeling of Transportation Risk for Hazardous Materials. *Operations Research* **46**(5), pp. 625-642, 1998.

[11] Halpern, J. Shortest Route with Time-Dependent Length of Edges and Limited Delay Possibilities in Nodes. *Zeitschrift fr Operations Research* **21**. pp. 117-124, 1977.

[12] Handler, G.Y. and I. Zang. A Dual Algorithm for the Constrained Shortest Path Problem.*Networks*. **10**. pp.293-310, 1980.

[13] Joksch, H.C. The Shortest Route Problem with Constraints. *Journal of Mathematical Analysis and Applications*. **14**. pp. 191-197, 1966.

[14] List G. F., P. B. Mirchandani, M. A. Turnquist and K. G. Zografos. Modeling and Analysis for Hazardous Materaials Transportation: Risk Analysis, Routing/Scheduling and Facility Location.*Transportation Science*. **31**. pp. 200-215, 1991.

[15] Nozick, L. N., G. F. List, M. A. Turnquist. Integrated Routing and Scheduling in Hazardous Materials Transportation. *Transportation Science* **31**(3), pp. 200-215, 1997.

[16] Orda, A. and R. Rom. Minimum Weigth Paths in Time-Dependent Networks. *Networks*. **21**. pp. 295-319, 1991.

[17] Riberio, C.C. and M. Minoux. A Heuristic Approach to Hard Constrained Shortest Path Problems. *Discrete Applied Mathematics*, **10**. pp. 125-137, 1989.

[18] Santos, L., J. Coutinho-Rodrigues and J.R. Current. An Improved Solution Algorithm for the Constrained Shortest Problem. *Transportation Resarch B*. **41**. pp 756-771, 2007.

[19] Schweitzer, L. Environmental Justice and Hazmat Transport: A Spatial Analysis in Southern California. *Transportation Research D.* **11**, pp. 408-421, 2006.

[20] Zhu X., and W. E. Wilhelm. Three-Stage Approaches for Optimizing Some Varitions of the Resource Constrained Shortest-Path Sub-Problem in a Column Generation Context. *Europen Journal of Operations Research.* **183**. pp 564-577, 2007.

[21] Alsleben S. Point & Polyline Tools v1.2. http://arcscripts.esri.com/details.asp?dbid=11694, as of January 12, 2009.

[22] General Directorate of Highways (Türkiye Cumhuriyeti Karayolları Genel Müdürlüğü). Accessed at http://www.kgm.gov.tr/ as of January 14, 2009.

[23] ESRI ArcView GIS. http://www.esri.com/software/arcgis/arcview/index.html as of January 5, 2009.

[24] EUR-LEX. Regulation (EC) No 561/2006 of the European Parliament and of the Council of 15 March 2006 on the harmonisation of certain social legislation relating to road transport and amending Council Regulations (EEC) No 3821/85 and (EC) No 2135/98 and repealing Council Regulation (EEC) No 3820/85 (Text with EEA relevance) - Declaration. Accessed at http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32006R0561:EN:HTML:NOT. Access to European Union Law as of as of January 14, 2009.

[25] Jenness J. Nearest Features v3.8.b. http://www.jennessent.com/arcview/nearest_features.htm, Jenness Enterprises. as of January 14, 2009.

[26] Jenness J. Path with Distances and Bearings v3.2.b. http://www.jennessent.com/arcview/path.htm, Jenness Enterprises. as of January 14, 2009.

[27] U.S. Department of Transportation. Hazardous Materials Information System, accessed at http://phmsa.dot.gov/hazmat/ as of March 30, 2009.

Appendix A

Breadth First Search

We use the Breadth-First Search algorithm to construct the initial path to be fed into the heuristic procedure. The pseudo-code of the procedure is given in Algorithm 5 [1] where, the LIST is traced as a first-in first-out queue in order to be a Breadth-First Search.

Algorithm 5 Breadth First Search

```
Unmark all nodes in N
Mark node s
pred(s) := 0
next := 1
order(s) := s
LIST :={s}
while LIST ≠ ∅ do
  Select a node i in LIST
  if Node i is incident to an admissible arc (i, j) then
    Mark node j
    pred(j):= i
    next := next+1
    order(j):= next
    Add node j to the LIST
  end if
  else Delete node i from LIST
end while
```

Appendix B

Dynamic Programming Formulation

The dynamic programming formulation, which is proposed by Erkut and Alp [8], is used in our heuristic algorithm as an adjunct so that the most promising paths are solved using this formulation. For detailed information about the formulation and the related parameters, see Erkut and Alp [8].

B.1 Notation

N	Number of nodes
m	Number of arcs
$f(y,t,v,w)$	Risk value of the minimum risk path from the origin to node y with a path duration of exactly t, uninterrupted driving time of v, duration of the current on duty period of w, and with waiting time at node y of zero
$f^*(y)$	Total risk of the minimum risk path from the origin to node y
$d_{ij}(t)$	Duration of arc (i,j) when entry time is t
r_{ij}	Risk experienced on arc (i,j) when entry time is t
T	Upper bound on the total duration of the path
D	Maximum uninterrupted driving time permissible
W	Maximum length of the on duty period
L_i	Lower bound on waiting at node i, if the waiting time is positive
U_i	Upper bound on waiting at node i, if the waiting time is positive
u_a	Arrival time from a node
u_d	Departure time from a node
u_r	Uninterrupted driving time upon arrival to node
u_w	Length of the on duty period upon arrival at a node
u_{sb}	Duration of a short break taken at a node
u_{lb}	Duration of a long break taken at a node
l_i	Lower bound on the short break given at node i

B.2 Formulation

$f^*(N) = \min_{t \le T,\ v \le D,\ w \le W} f(N,t,v,w)$ with
$f(1,t,0,0) = 0,\ \forall t$
$f(1,t,v,w) = \infty,\ \forall v > 0, w > 0$
$f(y,t,v,w) = \min_{\{x:(x,y)\in E\}} \min_{\{(u_a,u_d,u_r,u_w,u_{sb},u_{lb})\in F(x,y,t,v,w)\}} \{f(x,u_a,u_r,u_w) + r_{xy}(u_d)\}$
Where $F(x,y,t,v,w) = \{(u_a,u_d,u_r,u_w,u_{sb},u_{lb}) :$
$(u_{lb} = 0;\ u_{sb} = 0;\ u_d : u_d + d_{xy}(u_d) = t;\ u_a = u_d;\ u_r = v - d_{xy}(u_d);$
$u_w = w - d_{xy}(u_d))$
or
$(u_{lb} = 0;\ l_x \le u_{sb} \le W;\ u_d : u_d + d_{xy}(u_d) = t$ and $d_{xy}(u_d) = v;$
$u_a = u_d - u_{sb};\ 0 \le u_r \le D;\ u_w - u_{sb} - d_{xy}(u_d))$
or
$(L_x \le u_{lb} \le U_x;\ u_{sb} = 0;\ u_d : u_d + d_{xy}(u_d) = t\ and\ d_{xy}(u_d) = v\ and$
$d_{xy}(u_d) = w;\ u_a = u_d - u_{lb};\ 0 \le u_r \le D;\ 0 \le u_w \le W)\}$
$2 \le y \le N,\ 1 \le t \le T,\ 1 \le v \le D,\ 1 \le w \le W$

Appendix C

Numerical Test Results

In this section, we present the computational test results for each parameter setting.

Table C.1: The Computation Results of Network NLT

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
NLT 1	360	-	-	15.201	66,639.048	65,730	65,730	0
	360	WS	5	22.293	66,639.048	65,730	65,730	0
	360	CO	(30,3)	4.650	66,639.048	65,808.9	65,730	0.12
	576	-	-	424.470	36,590.715	27,613	27,613	0
	576	WS	3	359.787	36,590.715	27,613	27,613	0
	576	CO	(10,3)	159.035	36,590.715	27,613	27,613	0
	576	CO	(20,3)	76.082	36,590.715	27,613	27,613	0
	576	CO	(30,1)	38.939	37,248.099	28,515	27,613	3.26
	576	CO	(30,2)	33.655	36,775.872	28,369	27,613	2.73
	576	CO	(30,3)	70.752	36,590.715	27,613	27,613	0
	576	CO	(40,3)	61.286	36,590.715	27,613	27,613	0
	576	CO	(50,3)	60.979	36,590.715	27,613	27,613	0
	576	CO	(60,3)	60.144	36,590.715	27,613	27,613	0
	576	CO	(70,3)	60.136	36,590.715	27,613	27,613	0
NLT 2	360	-	-	6.893	18,567.486	18,567.486	18,567.486	0
	360	WS	5	5.498	18,567.486	18,567.486	18,567.486	0
	360	CO	(30,3)	1.404	18,567.486	18,567.486	18,567.486	0
	576	-	-	119.709	18,567.486	7,189	7,189	0
	576	WS	3	19.270	18,567.486	7,189	7,189	0
	576	CO	(10,3)	25.509	18,567.486	7,189	7,189	0
	576	CO	(20,3)	23.422	18,567.486	7,189	7,189	0
	576	CO	(30,1)	9.998	18,567.486	7,189	7,189	0
	576	CO	(30,2)	13.607	18,567.486	7,189	7,189	0
	576	CO	(30,3)	19.135	18,567.486	7,189	7,189	0
	576	CO	(40,3)	19.183	18,567.486	7,189	7,189	0
	576	CO	(50,3)	19.179	18,567.486	7,189	7,189	0
	576	CO	(60,3)	19.138	18,567.486	7,189	7,189	0
	576	CO	(70,3)	19.140	18,567.486	7,189	7,189	0

Table C.1: The Computation Results of Network NLT (continued)

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
NLT 3*	360	-	-	-	-	-	37,915	-
	360	WS	5	-	-	-	37,915	-
	360	CO	(30,3)	-	-	-	37,915	-
	576	-	-	36.579	22,510.912	17,244	17,244	0
	576	WS	3	24.126	22,510.912	17,244	17,244	0
	576	CO	(10,3)	16.982	22,510.912	17,244	17,244	0
	576	CO	(20,3)	16.850	22,510.912	17,244	17,244	0
	576	CO	(30,1)	4.82	23,495.455	20,182.2	17,244	17.03
	576	CO	(30,2)	9.242	22,510.912	17,244	17,244	0
	576	CO	(30,3)	16.918	22,510.912	17,244	17,244	0
	576	CO	(40,3)	16.890	22,510.912	17,244	17,244	0
	576	CO	(50,3)	16.897	22,510.912	17,244	17,244	0
	576	CO	(60,3)	16.897	22,510.912	17,244	17,244	0
	576	CO	(70,3)	16.882	22,510.912	17,244	17,244	0

*: The heuristic algorithm could not find any feasible solution for $T = 360$ case, since the minimum feasible trip time is 355 periods which is very close to 360.

Table C.2: The Computation Results of Sample Networks

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
Network 1	360	-	-	18.336	8,948	8,948	8,948	0
	360	WS	5	10.997	8,948	8,948	8,948	0
	360	CO	(30,3)	9.489	8,948	8,948	8,948	0
	576	-	-	82.007	6,143.775	4,692	2,260.44	107
	576	WS	3	65.767	6,143.775	4,692	2,260.44	107
	576	CO	(10,3)	43.716	6,143.775	4,692	2,260.44	107
	576	CO	(20,3)	37.597	6,143.775	4,692	2,260.44	107
	576	CO	(30,1)	2.522	6,143.775	2,260.44	2,260.44	0
	576	CO	(30,2)	6.678	6,143.775	4,692	2,260.44	107
	576	CO	(30,3)	40.184	6,143.775	4,692	2,260.44	107
	576	CO	(40,3)	13.972	6,143.775	4,692	2,260.44	107
	576	CO	(50,3)	12.019	6,143.775	4,692	2,260.44	107
	576	CO	(60,3)	12.020	6,143.775	4,692	2,260.44	107
	576	CO	(70,3)	12.019	6,143.775	4,692	2,260.44	107
Network 2	360	-	-	145.463	6,653.466	3,789.85	3,789.85	0
	360	WS	5	128.400	6,653.466	3,789.85	3,789.85	0
	360	CO	(30,3)	24.517	6,653.466	3,789.85	3,789.85	0
	576	-	-	553.509	6,653.466	1,909.84	1,909.84	0
	576	WS	3	327.452	6,653.466	1,909.84	1,909.84	0
	576	CO	(10,3)	180.837	6,653.466	1,909.84	1,909.84	0
	576	CO	(20,3)	117.883	6,653.466	1,909.84	1,909.84	0
	576	CO	(30,1)	112.791	6,653.466	1,909.84	1,909.84	0
	576	CO	(30,2)	36.465	6,653.466	1,909.84	1,909.84	0
	576	CO	(30,3)	93.495	6,653.466	1,909.84	1,909.84	0
	576	CO	(40,3)	102.190	6,653.466	1,909.84	1,909.84	0
	576	CO	(50,3)	67.760	6,653.466	1,909.84	1,909.84	0
	576	CO	(60,3)	66.93	6,653.466	1,909.84	1,909.84	0
	576	CO	(70,3)	66.979	6,653.466	1,909.84	1,909.84	0

Table C.2: The Computation Results of Sample Networks (continued)

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
Network 3	360	-	-	84.383	7,866	4,871.62	2,765.01	76.18
	360	WS	5	52.181	7,866	4,871.62	2,765.01	76.18
	360	CO	(30,3)	0.427	7,866	4,871.62	2,765.01	76.18
	576	-	-	337.983	7,866	2,574	1,662.59	54.81
	576	WS	3	1.120	7,866	2,574	1,662.59	54.81
	576	CO	(10,3)	57.821	7,866	2,574	1,662.59	54.81
	576	CO	(20,3)	53.038	7,866	2,574	1,662.59	54.81
	576	CO	(30,1)	0.022	7,866	2,574	1,662.59	54.81
	576	CO	(30,2)	0.133	7,866	2,574	1,662.59	54.81
	576	CO	(30,3)	1.344	7,866	2,574	1,662.59	54.81
	576	CO	(40,3)	1.329	7,866	2,574	1,662.59	54.81
	576	CO	(50,3)	1.473	7,866	2,574	1,662.59	54.81
	576	CO	(60,3)	1.475	7,866	2,574	1,662.59	54.81
	576	CO	(70,3)	1.246	7,866	2,574	1,662.59	54.81
Network 4	360	-	-	7.755	2,182.261	1,897.52	1,897.52	0
	360	WS	5	4.925	2,182.261	1,897.52	1,897.52	0
	360	CO	(30,3)	1.232	2,182.261	1,897.52	1,897.52	0
	576	-	-	33.225	2,182.261	1,897.52	1,897.52	0
	576	WS	3	3.819	2,182.261	1,897.52	1,897.52	0
	576	CO	(10,3)	10.274	2,182.261	1,897.52	1,897.52	0
	576	CO	(20,3)	5.830	2,182.261	1,897.52	1,897.52	0
	576	CO	(30,1)	1.096	2,182.261	1,897.52	1,897.52	0
	576	CO	(30,2)	2.795	2,182.261	1,897.52	1,897.52	0
	576	CO	(30,3)	4.799	2,182.261	1,897.52	1,897.52	0
	576	CO	(40,3)	4.756	2,182.261	1,897.52	1,897.52	0
	576	CO	(50,3)	4.250	2,182.261	1,897.52	1,897.52	0
	576	CO	(60,3)	3.818	2,182.261	1,897.52	1,897.52	0
	576	CO	(70,3)	3.824	2,182.261	1,897.52	1,897.52	0

Table C.2: The Computation Results of Sample Networks (continued)

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
Network 5	360	-	-	21.618	18,981	18,981	18,981	0
	360	WS	5	15.106	18,981	18,981	18,981	0
	360	CO	(30,3)	5.425	18,981	18,981	18,981	0
	576	-	-	95.558	11,858.886	5,412	5,240	3.28
	576	WS	3	110.348	11,858.886	5,412	5,240	3.28
	576	CO	(10,3)	55.307	11,858.886	5,412	5,240	3.28
	576	CO	(20,3)	31.595	11,858.886	5,412	5,240	3.28
	576	CO	(30,1)	1.242	11,858.886	5,412	5,240	3.28
	576	CO	(30,2)	15.640	11,858.886	5,412	5,240	3.28
	576	CO	(30,3)	24.309	11,858.886	5,412	5,240	3.28
	576	CO	(40,3)	20.073	11,858.886	5,412	5,240	3.28
	576	CO	(50,3)	19.190	11,858.886	5,412	5,240	3.28
	576	CO	(60,3)	18.514	11,858.886	5,412	5,240	3.28
	576	CO	(70,3)	18.514	11,858.886	5,412	5,240	3.28
Network 6	360	-	-	178.0191	1,629.186	774.628	774.628	0
	360	WS	5	109.237	1,629.186	774.628	774.628	0
	360	CO	(30,3)	14.489	1,629.186	774.628	774.628	0
	576	-	-	691.567	1,629.186	774.628	774.628	0
	576	WS	3	371.118	1,629.186	774.628	774.628	0
	576	CO	(10,3)	94.567	1,629.186	774.628	774.628	0
	576	CO	(20,3)	12.720	1,629.186	774.628	774.628	0
	576	CO	(30,1)	19.988	1,629.186	774.628	774.628	0
	576	CO	(30,2)	16.829	1,629.186	774.628	774.628	0
	576	CO	(30,3)	54.536	1,629.186	774.628	774.628	0
	576	CO	(40,3)	2.509	1,629.186	774.628	774.628	0
	576	CO	(50,3)	2.510	1,629.186	774.628	774.628	0
	576	CO	(60,3)	1.885	1,629.186	774.628	774.628	0
	576	CO	(70,3)	1.885	1,629.186	774.628	774.628	0

Table C.2: The Computation Results of Sample Networks (continued)

Network	T_{MAX}	**Method**	**Parameters** (λ, ϵ) **or** w	**CPU Time (in minutes)**	**Minimum Heuristic Risk**	**Optimal Risk of Heuristic**	**Optimal Risk**	**Gap (%)**
Network 7	360	-	-	37.125	3,067.502	779.631	779.631	0
	360	WS	5	67.169	3,067.502	779.631	779.631	0
	360	CO	(30,3)	0.189	3,067.502	779.631	779.631	0
	576	-	-	127.322	3,067.502	700.439	700.439	0
	576	WS	3	539.413	3,067.502	700.439	700.439	0
	576	CO	(10,3)	72.743	3,067.502	700.439	700.439	0
	576	CO	(20,3)	85.408	3,067.502	700.439	700.439	0
	576	CO	(30,1)	12.219	3,067.502	700.439	700.439	0
	576	CO	(30,2)	0.163	3,067.502	700.439	700.439	0
	576	CO	(30,3)	0.378	3,067.502	700.439	700.439	0
	576	CO	(40,3)	0.375	3,067.502	700.439	700.439	0
	576	CO	(50,3)	0.377	3,067.502	700.439	700.439	0
	576	CO	(60,3)	0.377	3,067.502	700.439	700.439	0
	576	CO	(70,3)	0.377	3,067.502	700.439	700.439	0